Multirotor UAV state estimation and localization

B(E)3M33MRS — Aerial Multi-Robot Systems

Ing. Tomáš Báča, Ph.D.

Multi-Robot Systems group, Faculty of Electrical Engineering
Czech Technical University in Prague

Lecture 3: UAV localization

2024-10-07

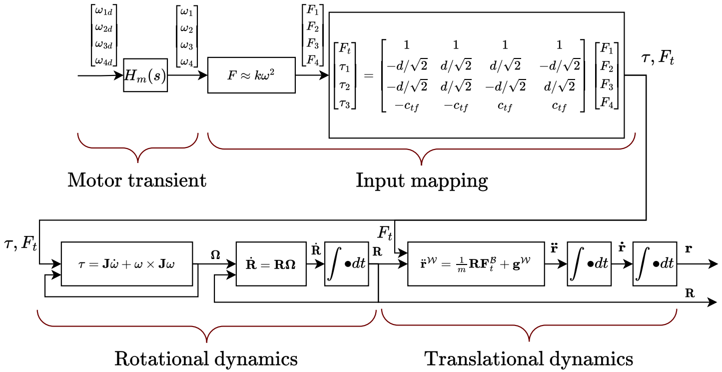
Multirotor UAV state estimation and localization
B(E)3M33MRS — Aerial Multi-Robot Systems
Ing. Tomáš Báča, Ph.D.
Multi-Robot Systems group, Faculty of Electrical Engineering
Czech Technical University in Prague

Obtaining UAV state for feedback control and autonomous navigation?

Closing the loop

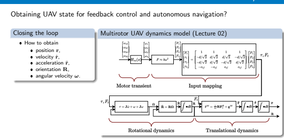
- How to obtain
  - position $\mathbf{r}$,
  - velocity $\dot{\mathbf{r}}$,
  - acceleration $\ddot{\mathbf{r}}$,
  - orientation $\mathbf{R}$,
  - angular velocity $\boldsymbol{\omega}$.

Multirotor UAV dynamics model (Lecture 02)



2024-10-07

Lecture 3: UAV localization
└─Introduction
└─Obtaining UAV state for feedback control and autonomous navigation?



Obtaining UAV state for feedback control and autonomous navigation?

Closing the loop

- How to obtain
  - position $\mathbf{r}$,
  - velocity $\dot{\mathbf{r}}$,
  - acceleration $\ddot{\mathbf{r}}$,
  - orientation $\mathbf{R}$,
  - angular velocity $\boldsymbol{\omega}$.

Problems with measurements

- Some system states can not be measured at all.
- Some system states can not be measured directly.
- Sometimes, the measurement rate is not high enough for control loop.
- Measurements tend to be noisy.
- Measurement precision might not be sufficient.

Obtaining UAV state for feedback control and autonomous navigation?
Closing the loop
How to obtain position r, velocity ṙ, acceleration r̈, orientation R, angular velocity ω.
Problems with measurements
Some system states can not be measured at all.
Some system states can not be measured directly.
Sometimes, the measurement rate is not high enough for control loop.
Measurements tend to be noisy.
Measurement precision might not be sufficient.

Obtaining UAV state for feedback control and autonomous navigation?

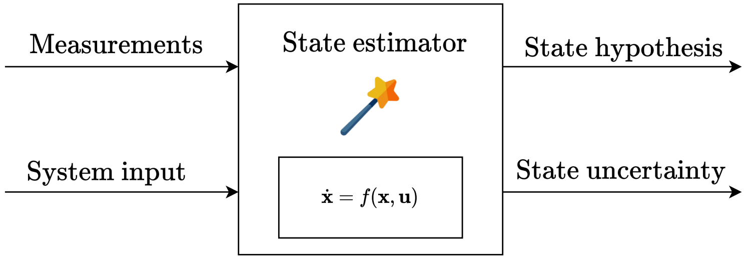
Closing the loop

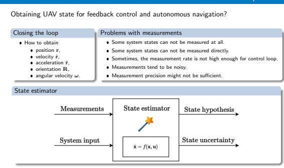
- How to obtain
  - position $\mathbf{r}$,
  - velocity $\dot{\mathbf{r}}$,
  - acceleration $\ddot{\mathbf{r}}$,
  - orientation $\mathbf{R}$,
  - angular velocity $\boldsymbol{\omega}$.

Problems with measurements

- Some system states can not be measured at all.
- Some system states can not be measured directly.
- Sometimes, the measurement rate is not high enough for control loop.
- Measurements tend to be noisy.
- Measurement precision might not be sufficient.

State estimator





- State estimator is often called state observer in the context of control systems.

Obtaining UAV state for feedback control and autonomous navigation?

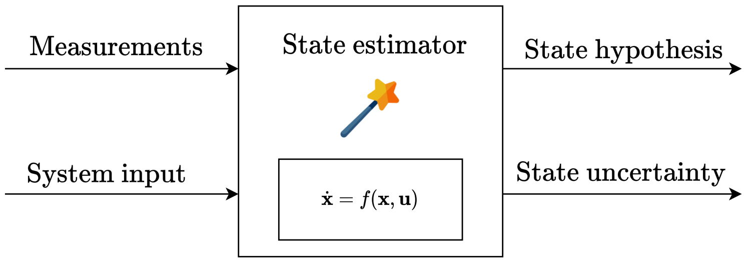
Closing the loop

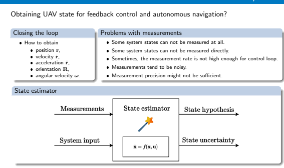
- How to obtain
  - position $\mathbf{r}$,
  - velocity $\dot{\mathbf{r}}$,
  - acceleration $\ddot{\mathbf{r}}$,
  - orientation $\mathbf{R}$,
  - angular velocity $\boldsymbol{\omega}$.

Problems with measurements

- Some system states can not be measured at all.
- Some system states can not be measured directly.
- Sometimes, the measurement rate is not high enough for control loop.
- Measurements tend to be noisy.
- Measurement precision might not be sufficient.

State estimator





- State estimator is often called state observer in the context of control systems.

Robot's current state

- Robot's belief of its current state.
- Probability Distribution Function (PDF), often multivariate normal distribution.

Robot's motion model

- Allows to predict robot's future state based on the current state and input.
- Transforms the *current state* distribution, based on input.

Robot's sensor model

- Allows to incorporate measurements into the current state probabilistically.
- Allows to create artificial measurements based on the world model and the robot's state.

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. Cambridge, Mass.: MIT Press, 2005

Probabilistic state estimation and localization

Robot's current state
- Robot's belief of its current state.
- Probability Distribution Function (PDF), often multivariate normal distribution.

Robot's motion model
- Allows to predict robot's future state based on the current state and input.
- Transforms the *current state* distribution, based on input.

Robot's sensor model
- Allows to incorporate measurements into the current state probabilistically.
- Allows to create artificial measurements based on the world model and the robot's state.

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. Cambridge, Mass.: MIT Press, 2005

- All three models are considered to be **stochastic**.

Robot's state is *complete*

- Robot's state at the time step k is all we need to predict the future:

$$\mathsf{p}(\mathbf{x}_{[k]}|\mathbf{x}_{[0:k-1]}, \mathbf{u}_{[1:k]}, \mathbf{z}_{[1:k-1]}) \rightarrow \mathsf{p}(\mathbf{x}_{[k]}|\mathbf{x}_{[k-1]}, \mathbf{u}_{[k-1]}). \quad (1)$$

- The measurement of robot's state is conditionally independent on the previous states:

$$\mathsf{p}(\mathbf{z}_{[k]}|\mathbf{x}_{[0:k-1]}, \mathbf{u}_{[1:k]}, \mathbf{z}_{[1:k-1]}) \rightarrow \mathsf{p}(\mathbf{z}_{[k]}|\mathbf{x}_{[k]}). \quad (2)$$

Probabilistic generative laws
Robot's state is *complete*
- Robot's state at the time step k is all we need to predict the future:
$$\mathsf{p}(\mathbf{x}_{[k]}|\mathbf{x}_{[0:k-1]}, \mathbf{u}_{[1:k]}, \mathbf{z}_{[1:k-1]}) \rightarrow \mathsf{p}(\mathbf{x}_{[k]}|\mathbf{x}_{[k-1]}, \mathbf{u}_{[k-1]}). \quad (1)$$
- The measurement of robot's state is conditionally independent on the previous states:
$$\mathsf{p}(\mathbf{z}_{[k]}|\mathbf{x}_{[0:k-1]}, \mathbf{u}_{[1:k]}, \mathbf{z}_{[1:k-1]}) \rightarrow \mathsf{p}(\mathbf{z}_{[k]}|\mathbf{x}_{[k]}). \quad (2)$$

- Gauss-Markov assumption states that the future and past states are decorrelated given the current state.

Probabilistic generative laws

Robot's state is *complete*

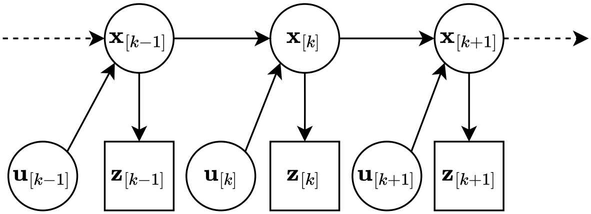
- Robot's state at the time step k is all we need to predict the future:

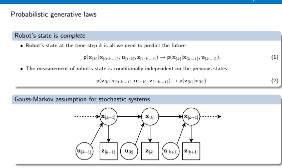
$$\mathsf{p}(\mathbf{x}_{[k]}|\mathbf{x}_{[0:k-1]}, \mathbf{u}_{[1:k]}, \mathbf{z}_{[1:k-1]}) \rightarrow \mathsf{p}(\mathbf{x}_{[k]}|\mathbf{x}_{[k-1]}, \mathbf{u}_{[k-1]}). \quad (1)$$

- The measurement of robot's state is conditionally independent on the previous states:

$$\mathsf{p}(\mathbf{z}_{[k]}|\mathbf{x}_{[0:k-1]}, \mathbf{u}_{[1:k]}, \mathbf{z}_{[1:k-1]}) \rightarrow \mathsf{p}(\mathbf{z}_{[k]}|\mathbf{x}_{[k]}). \quad (2)$$

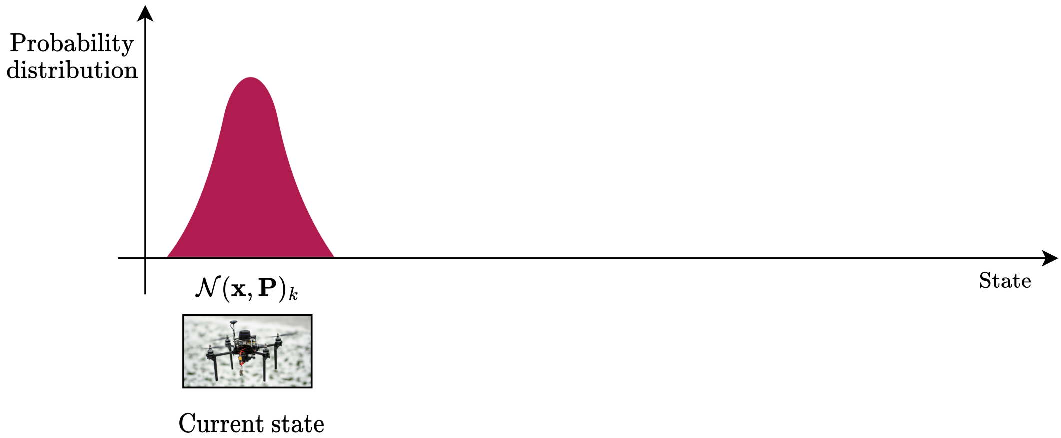
Gauss-Markov assumption for stochastic systems



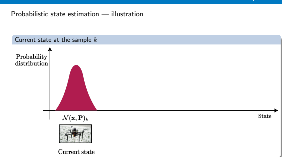


- Gauss-Markov assumption states that the future and past states are decorrelated given the current state.

Current state at the sample k

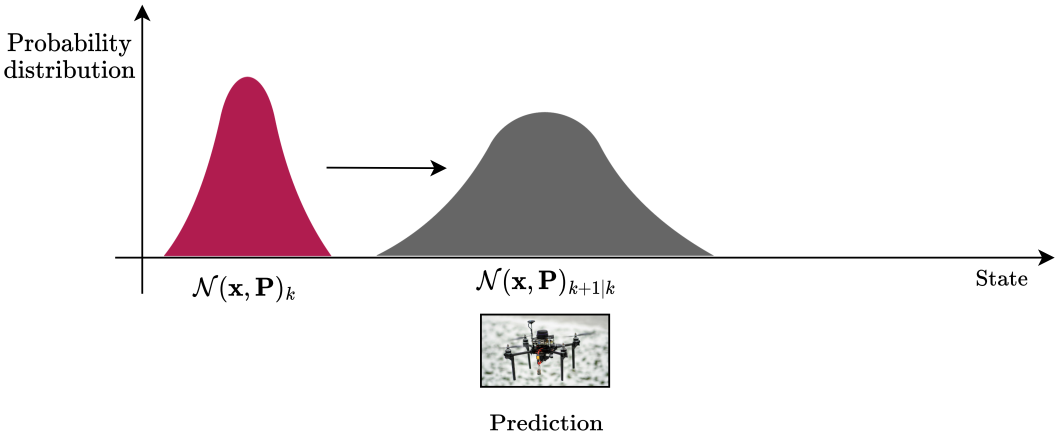


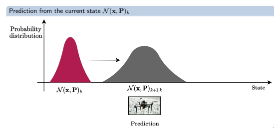
2024-10-07



- Current state is all we need to capture the past.
- Current state yields the prediction of the future state.
- The future states is measured.
- The prediction and the measurement are combined to form the estimate of the future state.

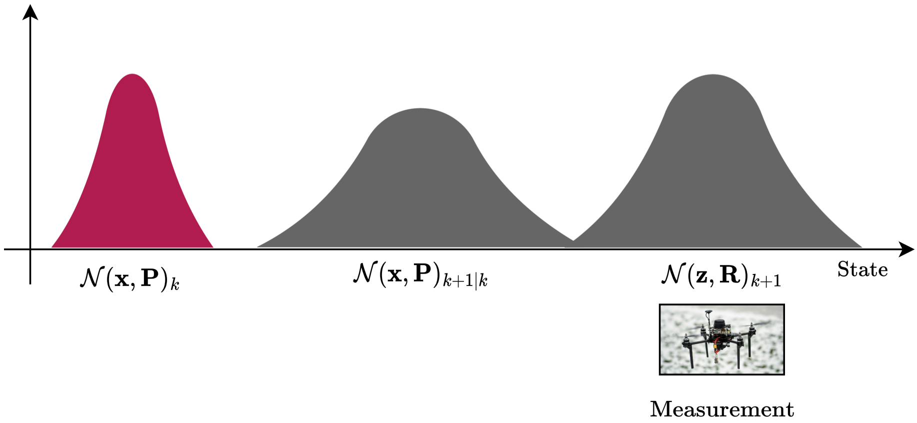
Prediction from the current state $\mathcal{N}(\mathbf{x}, \mathbf{P})_k$

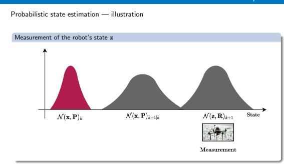




- Current state is all we need to capture the past.
- Current state yields the prediction of the future state.
- The future states is measured.
- The prediction and the measurement are combined to form the estimate of the future state.

Measurement of the robot's state $\mathbf{z}$

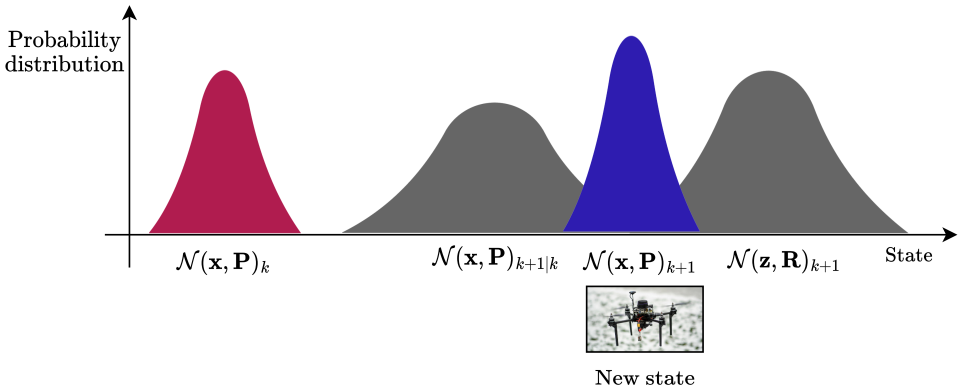


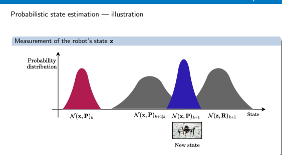


- Current state is all we need to capture the past.
- Current state yields the prediction of the future state.
- The future states is measured.
- The prediction and the measurement are combined to form the estimate of the future state.

Probabilistic state estimation — illustration

Measurement of the robot's state $\mathbf{z}$





- Current state is all we need to capture the past.
- Current state yields the prediction of the future state.
- The future states is measured.
- The prediction and the measurement are combined to form the estimate of the future state.

Linear Kalman Filter (LKF)

- Developed in $\approx$ 1960 at NASA.
- Optimal state estimator for linear models.
- Minimum Mean-Square Error estimator.

Models

- State: Multivariate Gaussian
- Sensor model: added noise $\mathcal{N}(\mathbf{0}, \mathbf{R})$
- Motion model: linear model with added noise $\mathcal{N}(\mathbf{0}, \mathbf{Q})$

Two-stage algorithm

- **Prediction**: propagation of robot's state and its uncertainty through the model.
- **Correction**: update of the robot's state and its uncertainty using measurements.

How to derive it?

- B3M35OFD, Estimation, filtering and detection

Linear Kalman Filter (LKF)

- Developed in $\approx$ 1960 at NASA.
- Optimal state estimator for linear models.
- Minimum Mean-Square Error estimator.

Models
- State: Multivariate Gaussian
- Sensor model: added noise $\mathcal{N}(\mathbf{0}, \mathbf{R})$
- Motion model: linear model with added noise $\mathcal{N}(\mathbf{0}, \mathbf{Q})$

Two-stage algorithm
- **Prediction**: propagation of robot's state and its uncertainty through the model.
- **Correction**: update of the robot's state and its uncertainty using measurements.

How to derive it?
- B3M35OFD, Estimation, filtering and detection

Estimator for the Linear Time Invariant (LTI) system

Discrete stochastic LTI System

- State at the time step k: $\mathbf{x}_{[k]} \in \mathbb{R}^n$.
- Input at the time step k: $\mathbf{u}_{[k]} \in \mathbb{R}^m$.

$$\mathbf{x}_{[k+1]} = \mathbf{A}\mathbf{x}_{[k]} + \mathbf{B}\mathbf{u}_{[k]} + \mathbf{w}_{[k]}, \tag{3}$$

where:

- System matrix: $\mathbf{A} \in \mathbb{R}^{n\times n}$,
- Input matrix: $\mathbf{B} \in \mathbb{R}^{n\times m}$,
- Process noise $\mathbf{w}_{[k]} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$,
- Process covariance matrix: $\mathbf{Q} \in \mathbb{R}^{n\times n}_{>0}$.

Measurement model

- Vector of measurements: $\mathbf{z} \in \mathbb{R}^p$.

$$\mathbf{z}_{[k]} = \mathbf{H}\mathbf{x}_{[k]} + \mathbf{v}_{[k]}, \tag{4}$$

where:

- Measurement-state mapping: $\mathbf{H} \in \mathbb{R}^{p\times n}$,
- Measurement noise: $\mathbf{v}_{[k]} \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$,
- Measurement noise covariance: $\mathbf{R} \in \mathbb{R}^{p\times p}_{>0}$.

Goal of the estimator

To estimate the tuple $\mathbf{x}^*_{[k]}, \mathbf{P}_{[k]}$, where

- $\mathbf{x}^*_{[k]}$ is the state vector estimate,
- $\mathbf{P}_{[k]} \in \mathbb{R}^{n\times n}$ is the state covariance.

Estimator for the Linear Time Invariant (LTI) system
Discrete stochastic LTI System
Measurement model
Goal of the estimator

Prediction

- Given $\mathbf{x}^*_{[k]}, \mathbf{P}_{[k]}, \mathbf{u}_{[k]}$:

$$\mathbf{x}^*_{[k+1]} = \mathbf{A}\mathbf{x}^*_{[k]} + \mathbf{B}\mathbf{u}_{[k]} \tag{5}$$

$$\mathbf{P}_{[k+1]} = \mathbf{A}\mathbf{P}_{[k]}\mathbf{A}^\intercal + \mathbf{Q} \tag{6}$$

Correction

- Given $\mathbf{P}_{[k]}$, calculate the *Kalman gain*:

$$\mathbf{K}_{[k]} = \mathbf{P}_{[k]}\mathbf{H}^\intercal \left(\mathbf{H}\mathbf{P}_{[k]}\mathbf{H}^\intercal + \mathbf{R}\right)^{-1}. \tag{7}$$

- Given $\mathbf{x}^*_{[k]}, \mathbf{P}_{[k]}, \mathbf{z}_{[k]}, \mathbf{K}_{[k]}$, update the state and its covariance:

$$\mathbf{x}^*_{[k]} := \mathbf{x}^*_{[k]} + \mathbf{K}_{[k]}\left(\mathbf{z}_{[k]} - \mathbf{H}\mathbf{x}^*_{[k]}\right), \tag{8}$$

$$\mathbf{P}_{[k]} := \left(\mathbf{I} - \mathbf{K}_{[k]}\mathbf{H}\right)\mathbf{P}_{[k]}, \tag{9}$$

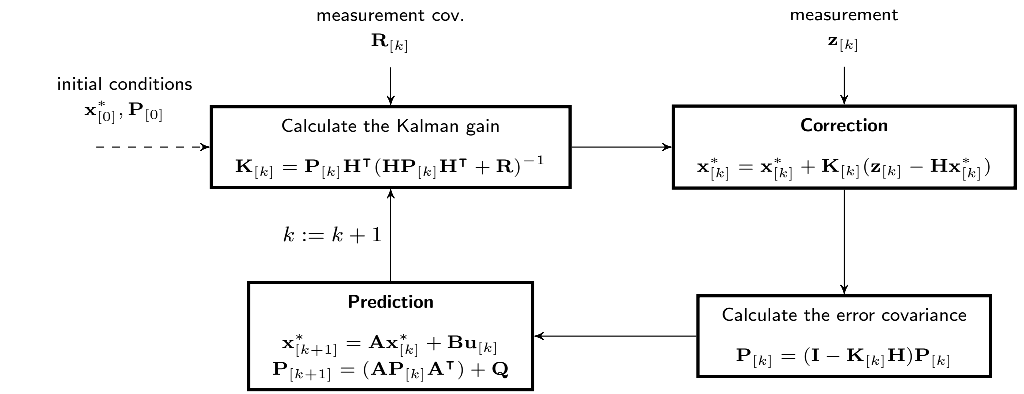
2024-10-07

Lecture 3: UAV localization
└─State estimators and Filters
  └─Linear Kalman Filter
    └─LKF

Linear Kalman Filter (LKF)
Prediction
Given $\mathbf{x}^*_{[k]}, \mathbf{P}_{[k]}, \mathbf{u}_{[k]}$:
$\mathbf{x}^*_{[k+1]} = \mathbf{A}\mathbf{x}^*_{[k]} + \mathbf{B}\mathbf{u}_{[k]}$ (5)
$\mathbf{P}_{[k+1]} = \mathbf{A}\mathbf{P}_{[k]}\mathbf{A}^\intercal + \mathbf{Q}$ (6)
Correction
Given $\mathbf{P}_{[k]}$, calculate the Kalman gain:
$\mathbf{K}_{[k]} = \mathbf{P}_{[k]}\mathbf{H}^\intercal \left(\mathbf{H}\mathbf{P}_{[k]}\mathbf{H}^\intercal + \mathbf{R}\right)^{-1}.$ (7)
Given $\mathbf{x}^*_{[k]}, \mathbf{P}_{[k]}, \mathbf{z}_{[k]}, \mathbf{K}_{[k]}$, update the state and its covariance:
$\mathbf{x}^*_{[k]} := \mathbf{x}^*_{[k]} + \mathbf{K}_{[k]}\left(\mathbf{z}_{[k]} - \mathbf{H}\mathbf{x}^*_{[k]}\right),$ (8)
$\mathbf{P}_{[k]} := \left(\mathbf{I} - \mathbf{K}_{[k]}\mathbf{H}\right)\mathbf{P}_{[k]},$ (9)

Linear Kalman Filter (LKF)

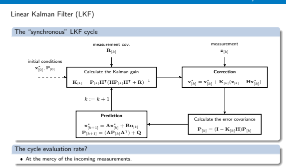
The "synchronous" LKF cycle



The cycle evaluation rate?

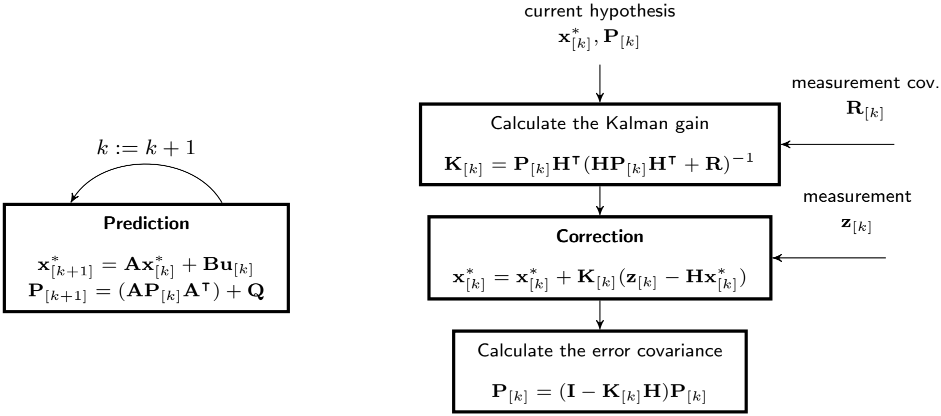
- At the mercy of the incoming measurements.

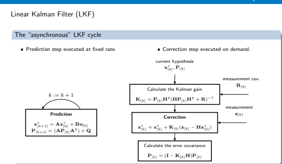
Lecture 3: UAV localization
State estimators and Filters
Linear Kalman Filter
Linear Kalman Filter (LKF)
2024-10-07



The "asynchronous" LKF cycle

- Prediction step executed at fixed rate.
- Correction step executed on demand.



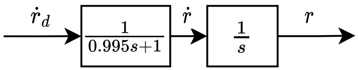


- More often, the prediction and correction happen asynchronously.
- Corrections can be even caused by variety of sources at different rate.
- The prediction step is often being evaluated at fixed rate. The state obtained at the prediction step is used for control.

Linear Kalman Filter (LKF) — Example

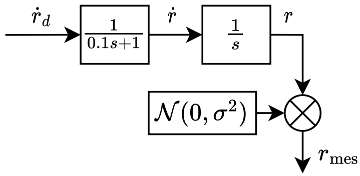
Example LTI system, $\Delta t = 0.01\,\mathrm{s}$

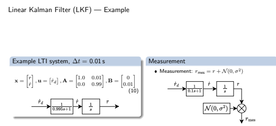
$$\mathbf{x} = \begin{bmatrix} r \\ \dot{r} \end{bmatrix}, \mathbf{u} = \begin{bmatrix} \dot{r}_d \end{bmatrix}, \mathbf{A} = \begin{bmatrix} 1.0 & 0.01 \\ 0.0 & 0.99 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 0 \\ 0.01 \end{bmatrix} \tag{10}$$



Measurement

- Measurement: $r_{\text{mes}} = r + \mathcal{N}(0, \sigma^2)$

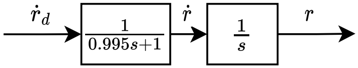




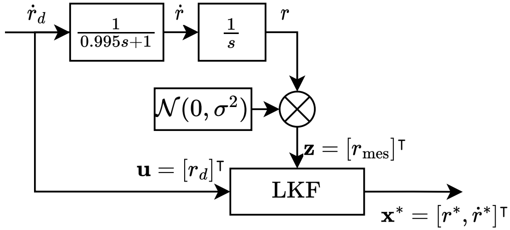
Linear Kalman Filter (LKF) — Example

Example LTI system, $\Delta t = 0.01\,\mathrm{s}$

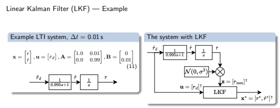
$$\mathbf{x} = \begin{bmatrix} r \\ \dot{r} \end{bmatrix}, \mathbf{u} = \begin{bmatrix} \dot{r}_d \end{bmatrix}, \mathbf{A} = \begin{bmatrix} 1.0 & 0.01 \\ 0.0 & 0.99 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 0 \\ 0.01 \end{bmatrix} \tag{11}$$



The system with LKF



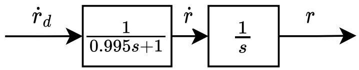
2024-10-07



Linear Kalman Filter (LKF) — Example

Example LTI system, $\Delta t = 0.01\,\mathrm{s}$

$$\mathbf{x} = \begin{bmatrix} r \\ \dot{r} \end{bmatrix}, \mathbf{u} = \begin{bmatrix} \dot{r}_d \end{bmatrix}, \mathbf{A} = \begin{bmatrix} 1.0 & 0.01 \\ 0.0 & 0.99 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 0 \\ 0.01 \end{bmatrix} \tag{11}$$

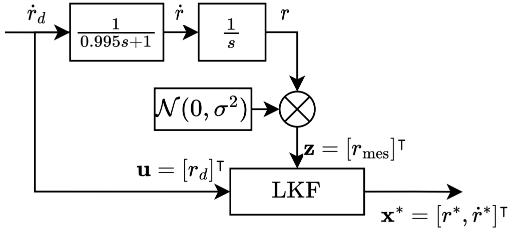


Measurement

- Measurement vector: $\mathbf{z} = [r_{\text{mes}}]^\intercal$.
- Measurement mapping:

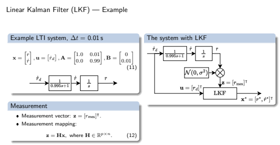
$$\mathbf{z} = \mathbf{H}\mathbf{x}, \text{ where } \mathbf{H} \in \mathbb{R}^{p\times n}. \tag{12}$$

The system with LKF



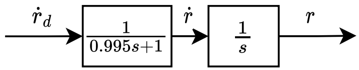
Lecture 3: UAV localization
└─State estimators and Filters
  └─Linear Kalman Filter
    └─Linear Kalman Filter (LKF) — Example

2024-10-07



Example LTI system, $\Delta t = 0.01\,\mathrm{s}$

$$\mathbf{x} = \begin{bmatrix} r \\ \dot{r} \end{bmatrix}, \mathbf{u} = \begin{bmatrix} \dot{r}_d \end{bmatrix}, \mathbf{A} = \begin{bmatrix} 1.0 & 0.01 \\ 0.0 & 0.99 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 0 \\ 0.01 \end{bmatrix} \tag{11}$$



Measurement

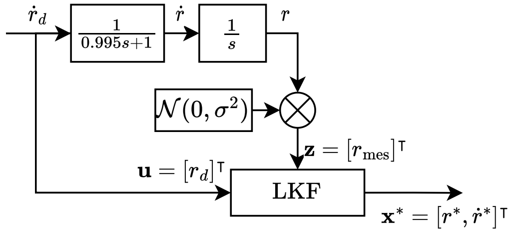
- Measurement vector: $\mathbf{z} = [r_{\text{mes}}]^\intercal$.
- Measurement mapping:

$$\mathbf{z} = \mathbf{H}\mathbf{x}, \text{ where } \mathbf{H} \in \mathbb{R}^{p \times n}. \tag{12}$$

- Measurement mapping matrix:

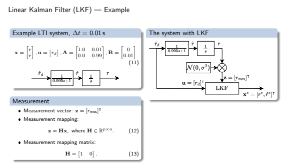
$$\mathbf{H} = \begin{bmatrix} 1 & 0 \end{bmatrix}. \tag{13}$$

The system with LKF



2024-10-07

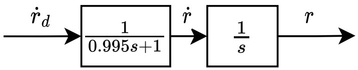
Lecture 3: UAV localization
└─State estimators and Filters
 └─Linear Kalman Filter
  └─Linear Kalman Filter (LKF) — Example



Linear Kalman Filter (LKF) — Example

Example LTI system, $\Delta t = 0.01\,\mathrm{s}$

$$\mathbf{x} = \begin{bmatrix} r \\ \dot{r} \end{bmatrix}, \mathbf{u} = \begin{bmatrix} \dot{r}_d \end{bmatrix}, \mathbf{A} = \begin{bmatrix} 1.0 & 0.01 \\ 0.0 & 0.99 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 0 \\ 0.01 \end{bmatrix} \tag{11}$$



Measurement

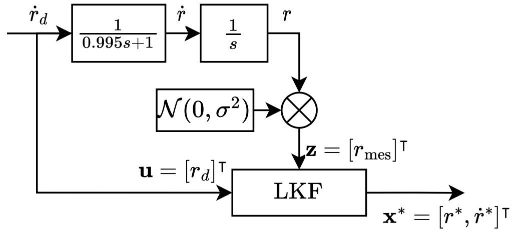
- Measurement vector: $\mathbf{z} = [r_{\text{mes}}]^\intercal$.
- Measurement mapping:

$$\mathbf{z} = \mathbf{H}\mathbf{x}, \text{ where } \mathbf{H} \in \mathbb{R}^{p \times n}. \tag{12}$$

- Measurement mapping matrix:

$$\mathbf{H} = \begin{bmatrix} 1 & 0 \end{bmatrix}. \tag{13}$$

The system with LKF



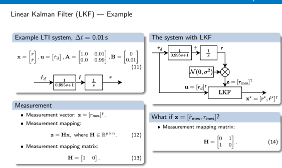
What if $\mathbf{z} = [\dot{r}_{\text{mes}}, r_{\text{mes}}]$?

- Measurement mapping matrix:

$$\mathbf{H} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}. \tag{14}$$

2024-10-07

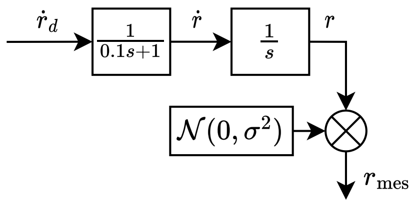
Lecture 3: UAV localization
└─State estimators and Filters
  └─Linear Kalman Filter
    └─Linear Kalman Filter (LKF) — Example

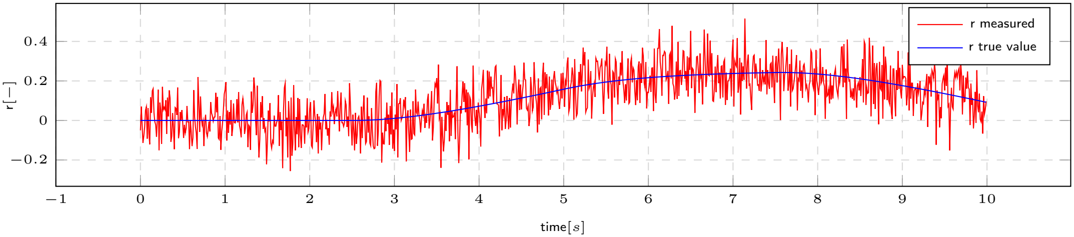


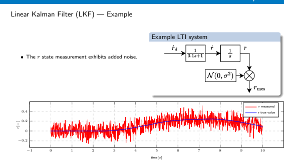
Linear Kalman Filter (LKF) — Example

- The r state measurement exhibits added noise.

Example LTI system



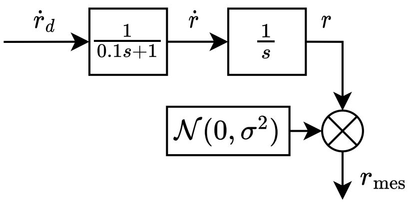


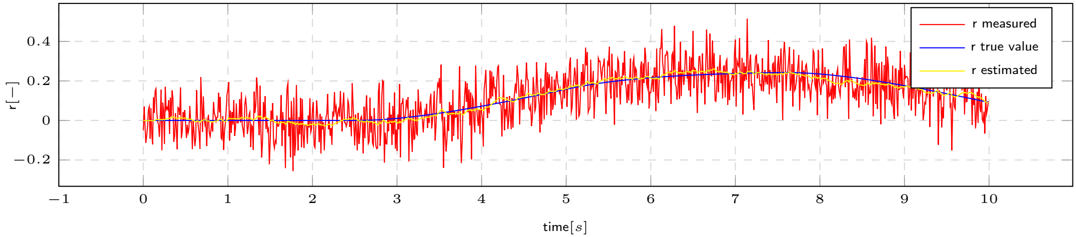


Linear Kalman Filter (LKF) — Example

- The r state measurement exhibits added noise.
- LKF can estimate the state while removing the noise.

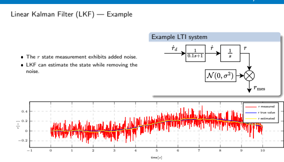
Example LTI system





2024-10-07

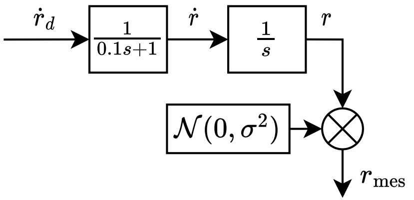
Lecture 3: UAV localization
└─State estimators and Filters
 └─Linear Kalman Filter
  └─Linear Kalman Filter (LKF) — Example

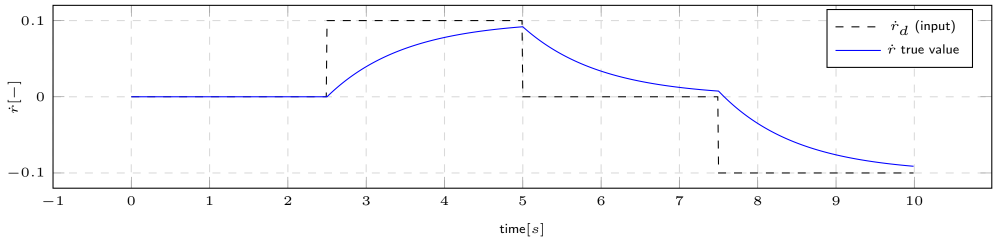


Linear Kalman Filter (LKF) — Example

- The $\dot{r}$ state is not a measured variable.

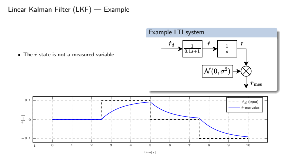
Example LTI system





2024-10-07

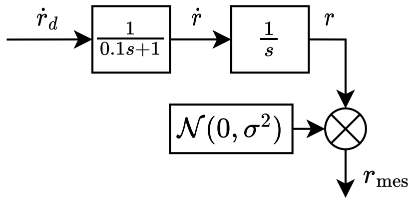
Lecture 3: UAV localization
└State estimators and Filters
 └Linear Kalman Filter
  └Linear Kalman Filter (LKF) — Example

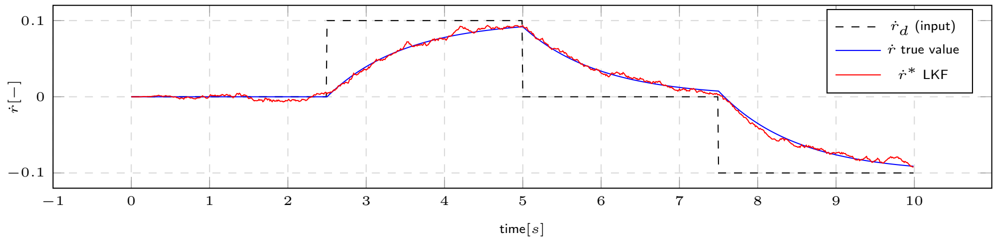


Linear Kalman Filter (LKF) — Example

- The $\dot{r}$ state is not a measured variable.
- LKF can estimate/observe the variable.

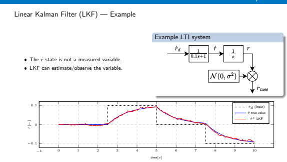
Example LTI system





2024-10-07

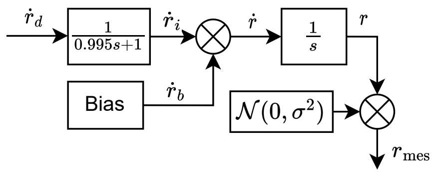
Lecture 3: UAV localization
└─State estimators and Filters
  └─Linear Kalman Filter
    └─Linear Kalman Filter (LKF) — Example



Estimating hidden states

- Hidden states are often used to model sensor biases.
- LKF can estimate them if they are observable.

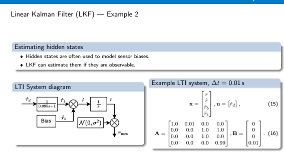
LTI System diagram



Example LTI system, $\Delta t = 0.01\,\text{s}$

$$\mathbf{x} = \begin{bmatrix} r \\ \dot{r} \\ \dot{r}_b \\ \dot{r}_i \end{bmatrix}, \mathbf{u} = \begin{bmatrix} \dot{r}_d \end{bmatrix}, \tag{15}$$

$$\mathbf{A} = \begin{bmatrix} 1.0 & 0.01 & 0.0 & 0.0 \\ 0.0 & 0.0 & 1.0 & 1.0 \\ 0.0 & 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.99 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0.01 \end{bmatrix}. \tag{16}$$



- Sensor bias estimation is heavily used to estimate nonzero offsets of sensors such as gyroscopes and accelerometers.
- Wind speed can be estimated as a bias in UAV acceleration.

What if we have a non-linear model?

- UAV Rotational dynamics.
- Ackermann vehicle.
- Differential car-like model.
- ... almost anything engineering-related in the real world.

Linearization?

- Needs an operation point.
- A single operation point is hard-to-find with most models.

Let's linearize more

- Extended Kalman Filter (EKF).
- De-facto standard in aviation and inertial navigation.

2024-10-07

Extended Kalman Filter (EKF) (EKF)

What if we have a non-linear model?
- UAV Rotational dynamics.
- Ackermann vehicle.
- Differential car-like model.
- ... almost anything engineering-related in the real world.

Linearization?
- Needs an operation point.
- A single operation point is hard-to-find with most models.

Let's linearize more
- Extended Kalman Filter (EKF).
- De-facto standard in aviation and inertial navigation.

Extended Kalman Filter (EKF)

Discrete stochastic system

- State at the time step k: $\mathbf{x}_{[k]} \in \mathbb{R}^n$.
- Input at the time step k: $\mathbf{u}_{[k]} \in \mathbb{R}^m$.

$$\mathbf{x}_{[k+1]} = f(\mathbf{x}_{[k]}, \mathbf{u}_{[k]}) + \mathbf{w}_{[k]}, \tag{17}$$

where:

- $f()$ is differentiable,
- Process noise $\mathbf{w}_{[k]} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$,
- Process covariance matrix: $\mathbf{Q} \in \mathbb{R}^{n\times n}_{>0}$.

Measurement model

- Vector of measurements: $\mathbf{z} \in \mathbb{R}^p$.

$$\mathbf{z}_{[k]} = h(\mathbf{x}_{[k]}) + \mathbf{v}_{[k]}, \tag{18}$$

where:

- Measurement-state mapping: $h() : \mathbb{R}^p \rightarrow \mathbb{R}^n$ is differentiable,
- Measurement noise: $\mathbf{v}_{[k]} \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$,
- Measurement noise covariance: $\mathbf{R} \in \mathbb{R}^{p\times p}_{>0}$.

Extended Kalman Filter (EKF)
Discrete stochastic system
State at the time step k: $\mathbf{x}_{[k]} \in \mathbb{R}^n$.
Input at the time step k: $\mathbf{u}_{[k]} \in \mathbb{R}^m$.
$\mathbf{x}_{[k+1]} = f(\mathbf{x}_{[k]}, \mathbf{u}_{[k]}) + \mathbf{w}_{[k]}$, (17)
where:
$f()$ is differentiable,
Process noise $\mathbf{w}_{[k]} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$,
Process covariance matrix: $\mathbf{Q} \in \mathbb{R}^{n\times n}_{>0}$.
Measurement model
Vector of measurements: $\mathbf{z} \in \mathbb{R}^p$.
$\mathbf{z}_{[k]} = h(\mathbf{x}_{[k]}) + \mathbf{v}_{[k]}$, (18)
where:
Measurement-state mapping: $h() : \mathbb{R}^p \rightarrow \mathbb{R}^n$ is differentiable,
Measurement noise: $\mathbf{v}_{[k]} \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$,
Measurement noise covariance: $\mathbf{R} \in \mathbb{R}^{p\times p}_{>0}$.

Extended Kalman Filter (EKF)

Prediction

- Given $\mathbf{x}^*_{[k]}, \mathbf{P}_{[k]}, \mathbf{u}_{[k]}$:

$$\mathbf{x}^*_{[k+1]} = f(\mathbf{x}^*_{[k]}, \mathbf{u}_{[k]}) \tag{19}$$

$$\mathbf{P}_{[k+1]} = \mathbf{F}_{[k]} \mathbf{P}_{[k]} \mathbf{F}^\intercal_{[k]} + \mathbf{Q}, \tag{20}$$

where

$$\mathbf{F}_{[k]} = \left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\mathbf{x}^*_{[k]}, \mathbf{u}_{[k]}} \tag{21}$$

is the Jacobian of $f()$ evaluated at the $\mathbf{x}^*_{[k]}, \mathbf{u}_{[k]}$.

Correction

- Given $\mathbf{P}_{[k]}$, calculate the *Kalman gain*:

$$\mathbf{K}_{[k]} = \mathbf{P}_{[k]} \mathbf{H}^\intercal_{[k]} \left(\mathbf{H}_{[k]} \mathbf{P}_{[k]} \mathbf{H}^\intercal_{[k]} + \mathbf{R} \right)^{-1}. \tag{22}$$

- Given $\mathbf{x}^*_{[k]}, \mathbf{P}_{[k]}, \mathbf{z}_{[k]}, \mathbf{K}_{[k]}$, update the state and its covariance:

$$\mathbf{x}^*_{[k]} := \mathbf{x}^*_{[k]} + \mathbf{K}_{[k]} \left(\mathbf{z}_{[k]} - h(\mathbf{x}^*_{[k]}) \right), \tag{23}$$

$$\mathbf{P}_{[k]} := \left(\mathbf{I} - \mathbf{K}_{[k]} \mathbf{H}_{[k]} \right) \mathbf{P}_{[k]}, \tag{24}$$

where

$$\mathbf{H}_{[k]} = \left.\frac{\partial h}{\partial \mathbf{x}}\right|_{\mathbf{x}^*_{[k]}} \tag{25}$$

is the Jacobian of $h()$ evaluated at the $\mathbf{x}^*_{[k]}$.

Extended Kalman Filter (EKF)

Prediction

Given $\mathbf{x}^*_{[k]}, \mathbf{P}_{[k]}, \mathbf{u}_{[k]}$: $\mathbf{x}^*_{[k+1]} = f(\mathbf{x}^*_{[k]}, \mathbf{u}_{[k]})$ (19) $\mathbf{P}_{[k+1]} = \mathbf{F}_{[k]} \mathbf{P}_{[k]} \mathbf{F}^\intercal_{[k]} + \mathbf{Q}$, (20) where $\mathbf{F}_{[k]} = \left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\mathbf{x}^*_{[k]}, \mathbf{u}_{[k]}}$ (21) is the Jacobian of $f()$ evaluated at the $\mathbf{x}^*_{[k]}, \mathbf{u}_{[k]}$.

Correction

Given $\mathbf{P}_{[k]}$, calculate the *Kalman gain*: $\mathbf{K}_{[k]} = \mathbf{P}_{[k]} \mathbf{H}^\intercal_{[k]} \left(\mathbf{H}_{[k]} \mathbf{P}_{[k]} \mathbf{H}^\intercal_{[k]} + \mathbf{R} \right)^{-1}$. (22) Given $\mathbf{x}^*_{[k]}, \mathbf{P}_{[k]}, \mathbf{z}_{[k]}, \mathbf{K}_{[k]}$, update the state and its covariance: $\mathbf{x}^*_{[k]} := \mathbf{x}^*_{[k]} + \mathbf{K}_{[k]} \left(\mathbf{z}_{[k]} - h(\mathbf{x}^*_{[k]}) \right)$, (23) $\mathbf{P}_{[k]} := \left(\mathbf{I} - \mathbf{K}_{[k]} \mathbf{H}_{[k]} \right) \mathbf{P}_{[k]}$, (24) where $\mathbf{H}_{[k]} = \left.\frac{\partial h}{\partial \mathbf{x}}\right|_{\mathbf{x}^*_{[k]}}$ (25) is the Jacobian of $h()$ evaluated at the $\mathbf{x}^*_{[k]}$.

EKF Properties

- Optimality? no

Extended Kalman Filter (EKF) — properties
EKF Properties
- Optimality? no

EKF Properties

- Optimality? no
- Stability? not guaranteed

Lecture 3: UAV localization
State estimators and Filters
Extended Kalman Filter
Extended Kalman Filter (EKF) — properties
2024-10-07

Extended Kalman Filter (EKF) — properties
EKF Properties
- Optimality? no
- Stability? not guaranteed

EKF Properties

- Optimality? no
- Stability? not guaranteed
- Ease of use? far from it

Extended Kalman Filter (EKF) — properties
EKF Properties
- Optimality? no
- Stability? not guaranteed
- Ease of use? far from it

Extended Kalman Filter (EKF) — properties

EKF Properties

- Optimality? no
- Stability? not guaranteed
- Ease of use? far from it

EKF Problems

- $f()$, and $h()$ needs to be differentiable.
- $f()$, and $h()$ are linearized *blindly* in each state.
- EKF is sensitive to model inaccuracies.
- EKF is sensitive to poor initialization.

Extended Kalman Filter (EKF) — properties
EKF Properties
- Optimality? no
- Stability? not guaranteed
- Ease of use? far from it
EKF Problems
- $f()$, and $h()$ needs to be differentiable.
- $f()$, and $h()$ are linearized *blindly* in each state.
- EKF is sensitive to model inaccuracies.
- EKF is sensitive to poor initialization.

EKF Properties

- Optimality? no
- Stability? not guaranteed
- Ease of use? far from it

EKF Problems

- $f()$, and $h()$ needs to be differentiable.
- $f()$, and $h()$ are linearized *blindly* in each state.
- EKF is sensitive to model inaccuracies.
- EKF is sensitive to poor initialization.

EKF mind-set problem

- How EKF deals with non-linearity? EKF works with the **original state Probability Distribution Function** (PDF) and a **degraded model** description.
- What about we swap it around? Let's transform a **degraded state PDF** through the **original model**.

Extended Kalman Filter (EKF) — properties
EKF Properties
- Optimality? no
- Stability? not guaranteed
- Ease of use? far from it
EKF Problems
- $f()$, and $h()$ needs to be differentiable.
- $f()$, and $h()$ are linearized *blindly* in each state.
- EKF is sensitive to model inaccuracies.
- EKF is sensitive to poor initialization.
EKF mind-set problem
- How EKF deals with non-linearity? EKF works with the **original state Probability Distribution Function** (PDF) and a **degraded model** description.
- What about we swap it around? Let's transform a **degraded state PDF** through the **original model**.

Unscented Kalman Filter

- Published in early 2000s by Uhlmann et al.
- Uses the full nonlinear model $f(), h()$.
- Does not linearize, therefore, $f(), h()$ can be arbitrary.
- More *elegant solution* than EKF.
- More *robust* than EKF.

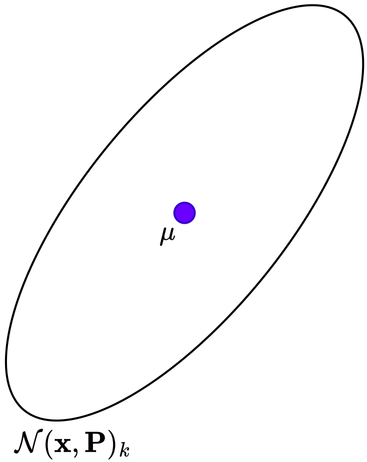
[2] E. A. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," in *Adaptive Systems for Signal Processing, Communications, and Control Symposium, IEEE*, IEEE, 2000, pp. 153–158

Unscented Kalman Filter (UKF)

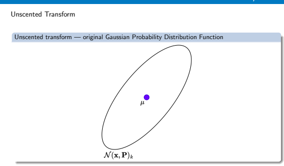
Unscented Kalman Filter
- Published in early 2000s by Uhlmann et al.
- Uses the full nonlinear model $f(), h()$.
- Does not linearize, therefore, $f(), h()$ can be arbitrary.
- More *elegant solution* than EKF.
- More *robust* than EKF.

[2] E. A. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," in *Adaptive Systems for Signal Processing, Communications, and Control Symposium, IEEE*, IEEE, 2000, pp. 153–158

Unscented transform — original Gaussian Probability Distribution Function

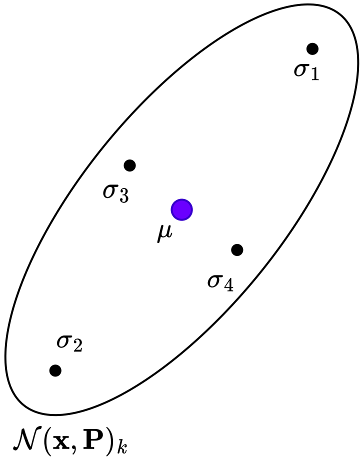


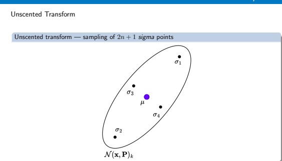
2024-10-07



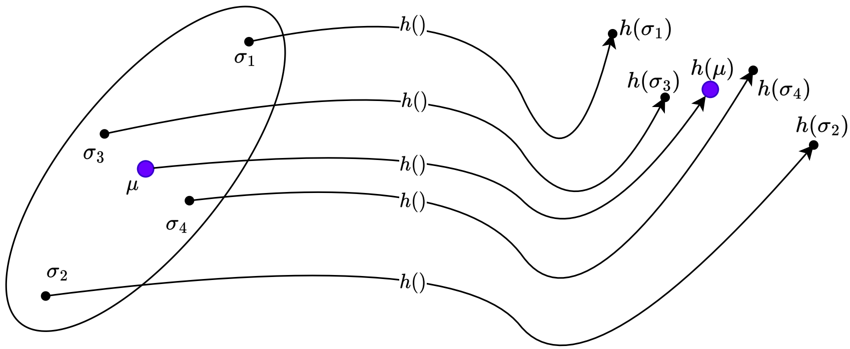
Unscented Transform

Unscented transform — sampling of $2n + 1$ *sigma* points

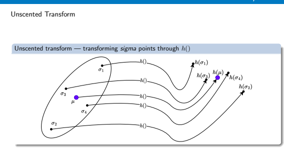




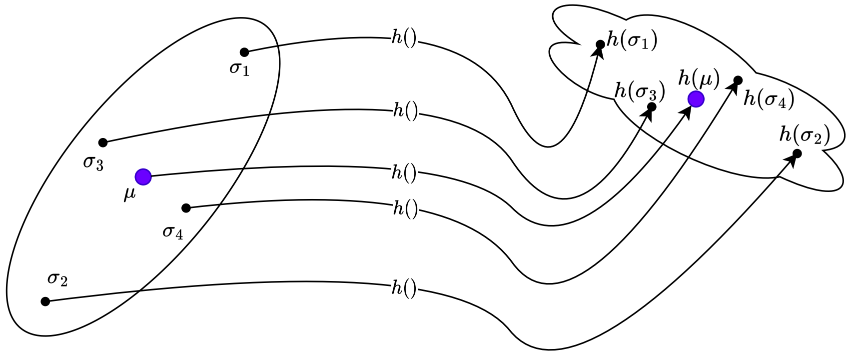
Unscented transform — transforming *sigma* points through $h()$



Lecture 3: UAV localization
State estimators and Filters
Unscented Kalman Filter
Unscented Transform
2024-10-07

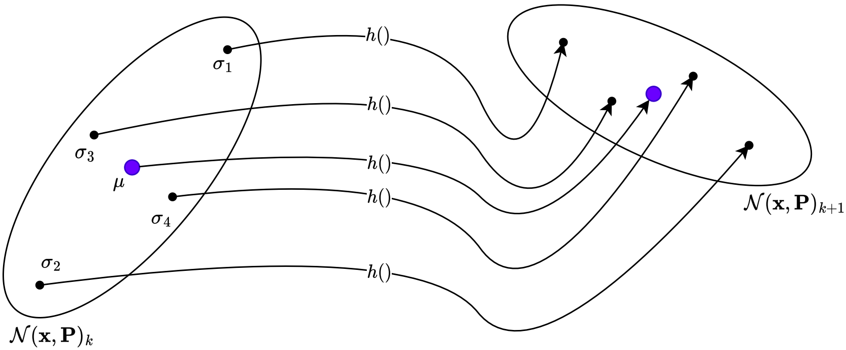


Unscented transform — transforming *sigma* points through $h()$

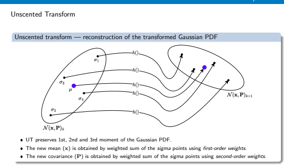


Lecture 3: UAV localization
State estimators and Filters
Unscented Kalman Filter
Unscented Transform

Unscented transform — reconstruction of the transformed Gaussian PDF



- UT preserves 1st, 2nd and 3rd moment of the Gaussian PDF.
- The new mean $(\mathbf{x})$ is obtained by weighted sum of the *sigma* points using *first-order weights*.
- The new covariance $(\mathbf{P})$ is obtained by weighted sum of the *sigma* points using *second-order weights*.



Unscented Kalman Filter (UKF) — Algorithm

Prediction step

1. Calculate the *sigma* points for $\mathbf{x}^*_{[k]}, \mathbf{P}_{[k]}$.
2. Propagate the *sigma* points through $f()$.
3. Reconstruct $\mathbf{x}^*_{[k+1]}, \mathbf{P}_{[k+1]}$

Correction step

1. Calculate the *sigma* points for $\mathbf{x}^*_{[k]}, \mathbf{P}_{[k]}$.
2. Propagate the *sigma* points through $h()$ to obtain the expected measurement $\mathbf{z}^*$.
3. Reconstruct the mean and covariance of the expected measurement.
4. Calculate cross-covariance between the measurement $\mathbf{z}$ and the expected measurement $\mathbf{z}^*$.
5. Calculate the Kalman gain using the cross-covariance.
6. Update the mean and covariance $\mathbf{x}^*, \mathbf{P}$.

[2] E. A. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," in *Adaptive Systems for Signal Processing, Communications, and Control Symposium, IEEE*, IEEE, 2000, pp. 153–158

Lecture 3: UAV localization
└─State estimators and Filters
└─Unscented Kalman Filter
└─Unscented Kalman Filter (UKF) — Algorithm

2024-10-07

Unscented Kalman Filter (UKF) — Algorithm
Prediction step
1. Calculate the sigma points for $\mathbf{x}^*_{[k]}, \mathbf{P}_{[k]}$.
2. Propagate the sigma points through $f()$.
3. Reconstruct $\mathbf{x}^*_{[k+1]}, \mathbf{P}_{[k+1]}$
Correction step
1. Calculate the sigma points for $\mathbf{x}^*_{[k]}, \mathbf{P}_{[k]}$.
2. Propagate the sigma points through $h()$ to obtain the expected measurement $\mathbf{z}^*$.
3. Reconstruct the mean and covariance of the expected measurement.
4. Calculate cross-covariance between the measurement $\mathbf{z}$ and the expected measurement $\mathbf{z}^*$.
5. Calculate the Kalman gain using the cross-covariance.
6. Update the mean and covariance $\mathbf{x}^*, \mathbf{P}$.
[2] E. A. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," in Adaptive Systems for Signal Processing, Communications, and Control Symposium, IEEE, IEEE, 2000, pp. 153–158

UKF Properties

- Optimality? still no

Unscented Kalman Filter (UKF) — properties
UKF Properties
- Optimality? still no

Unscented Kalman Filter (UKF) — properties

UKF Properties

- Optimality? still no
- Stability? still not guaranteed, but much better than EKF

Unscented Kalman Filter (UKF) — properties
UKF Properties
- Optimality? still no
- Stability? still not guaranteed, but much better than EKF

UKF Properties

- Optimality? still no
- Stability? still not guaranteed, but much better than EKF
- Ease of use? yes

2024-10-07
Lecture 3: UAV localization
State estimators and Filters
Unscented Kalman Filter
Unscented Kalman Filter (UKF) — properties

Unscented Kalman Filter (UKF) — properties
UKF Properties
- Optimality? still no
- Stability? still not guaranteed, but much better than EKF
- Ease of use? yes

UKF Properties

- Optimality? still no
- Stability? still not guaranteed, but much better than EKF
- Ease of use? yes

UKF Benefits

- No need to derive Jacobians.
- $h()$ and $f()$ can be arbitrary.
- Only the implementation of $h()$ and $f()$ needs to be supplied.

UKF Problems

- Does not have many.
- Mathematical soundness of operations needs to be checked (square rooting of $\mathbf{P}$).

2024-10-07
Lecture 3: UAV localization
└─State estimators and Filters
└─Unscented Kalman Filter
└─Unscented Kalman Filter (UKF) — properties

Unscented Kalman Filter (UKF) — properties

UKF Properties
- Optimality? still no
- Stability? still not guaranteed, but much better than EKF
- Ease of use? yes

UKF Benefits
- No need to derive Jacobians.
- $h()$ and $f()$ can be arbitrary.
- Only the implementation of $h()$ and $f()$ needs to be supplied.

UKF Problems
- Does not have many.
- Mathematical soundness of operations needs to be checked (square rooting of $\mathbf{P}$).

Unscented Kalman Filter (UKF) — example

State vector

The state vector is

$$\mathbf{x} = \left[\mathbf{r}^\intercal, \dot{\mathbf{r}}^\intercal, \eta, \dot{\eta}\right]^\intercal, \tag{26}$$

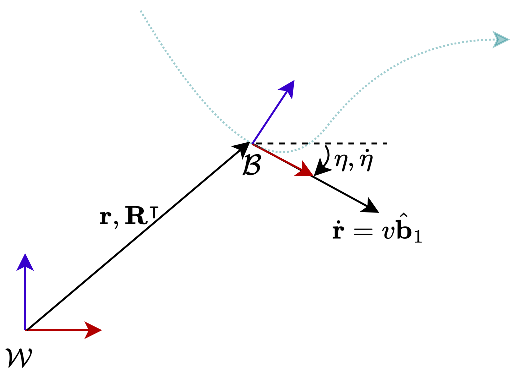
where

- $\mathbf{r}^{\mathcal{W}}$: the 2D position in the world frame,
- $\dot{\mathbf{r}}^{\mathcal{B}}$: the 2D velocity in the body frame,
- η: the heading,
- $\dot{\eta}$: the heading rate.

Measurement vector

$$\mathbf{z} = \left[\mathbf{r}^\intercal, \dot{\mathbf{r}}^\intercal, \dot{\eta}\right]^\intercal \tag{27}$$

System illustration



2024-10-07
Lecture 3: UAV localization
State estimators and Filters
Unscented Kalman Filter
Unscented Kalman Filter (UKF) — example

Unscented Kalman Filter (UKF) — example

Motion model, Δt

$$\mathbf{x}_{[k+1]} = \begin{bmatrix} \mathbf{r} \\ \dot{\mathbf{r}} \\ \eta \\ \dot{\eta} \end{bmatrix}_{[k+1]} = \begin{bmatrix} \mathbf{r}_{[k]} + \Delta t \mathbf{R}_{[k]} \dot{\mathbf{r}}_{[k]} \\ \dot{\mathbf{r}}_{[k]} \\ \eta_{[k]} + \Delta t \dot{\eta}_{[k]} \\ \dot{\eta}_{[k]} \end{bmatrix}, \quad (26)$$

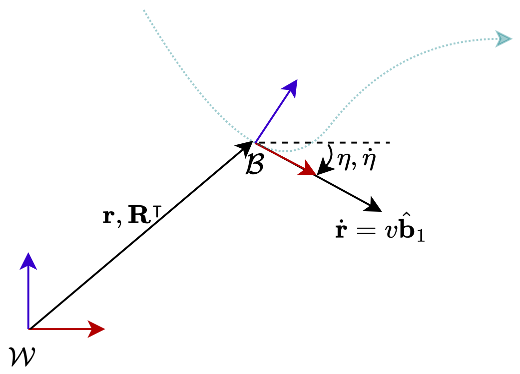
where

$$\mathbf{R}_{[k]} = \begin{bmatrix} \cos \eta_{[k]} & -\sin \eta_{[k]} \\ \sin \eta_{[k]} & \cos \eta_{[k]} \end{bmatrix}. \quad (27)$$

Observation model

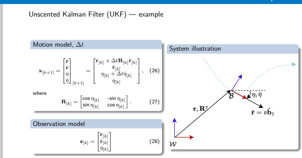
$$\mathbf{z}_{[k]} = \begin{bmatrix} \mathbf{r}_{[k]} \\ \dot{\mathbf{r}}_{[k]} \\ \dot{\eta}_{[k]} \end{bmatrix} \quad (28)$$

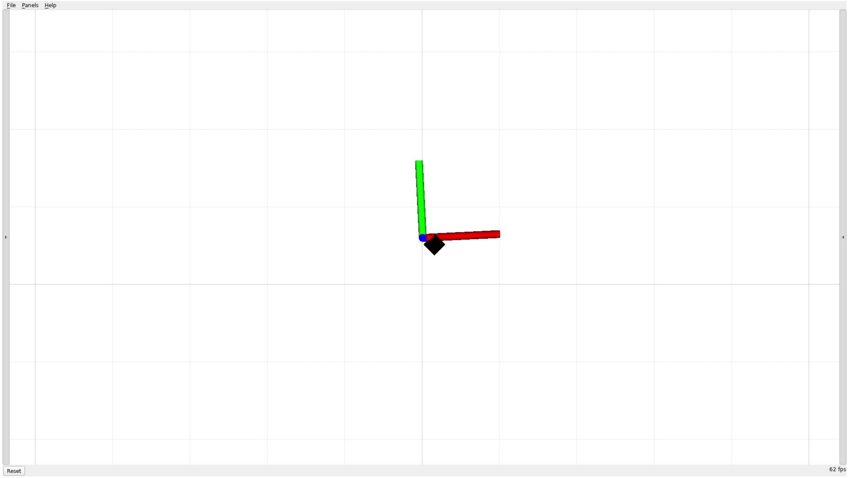
System illustration



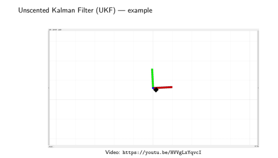
Lecture 3: UAV localization
└─State estimators and Filters
└─Unscented Kalman Filter
└─Unscented Kalman Filter (UKF) — example

2024-10-07





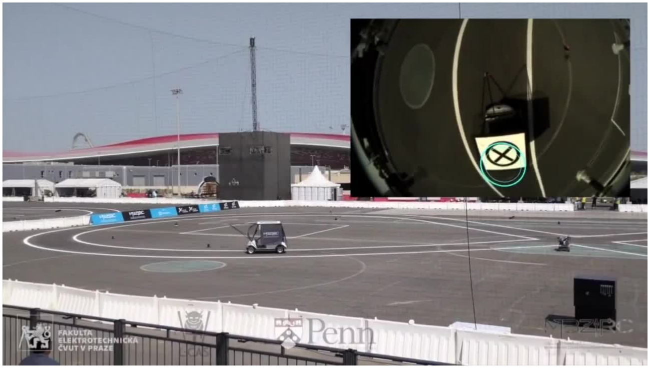
Video: `https://youtu.be/HVVgLxYqvcI`



Unscented Kalman Filter (UKF) — example 2

State estimation of a car

- Nonlinear car-like model (similar to the previous example).
- Unscented Kalman Filter.
- Car observed by a single camera.

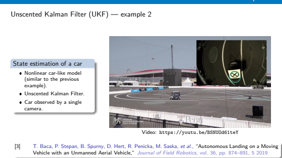


Video: `https://youtu.be/BSNUOd61teY`

[3] T. Baca, P. Stepan, B. Spurny, D. Hert, R. Penicka, M. Saska, *et al.*, "Autonomous Landing on a Moving Vehicle with an Unmanned Aerial Vehicle," *Journal of Field Robotics*, vol. 36, pp. 874–891, 5 2019

2024-10-07

Lecture 3: UAV localization
└─State estimators and Filters
└─Unscented Kalman Filter
└─Unscented Kalman Filter (UKF) — example 2



Rotation+translation Dynamics model

$$\boldsymbol{\tau} = \mathbf{J}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \mathbf{J}\boldsymbol{\omega} \tag{29}$$

$$\dot{\mathbf{R}} = \mathbf{R}\boldsymbol{\Omega} \tag{30}$$

$$\ddot{\mathbf{r}}^{\mathcal{W}} = \frac{1}{m}\mathbf{R}\mathbf{F}_t^{\mathcal{B}} + \mathbf{g}^{\mathcal{W}} \tag{31}$$

What do we need?

- Angular velocity: $\boldsymbol{\omega}^{\mathcal{B}}$.
- Orientation: $\mathbf{R}$.

State estimator

- a) Complementary filter
- b) EKF
- Often based on quaternions.
- Bias estimation for all the sensors.

UAV Onboard Sensors

- Gyroscope:
  - 3-axis MEMS.
  - **Measured intrinsic angular rate**.
  - Sufficient for attitude rate control.
- Accelerometer:
  - 3-axis MEMS.
  - **Measures proper acceleration**.
  - Gravity model might be needed for precise navigation.
- "Magnetometer":
  - 3-axis
  - **Measures external magnetic field**.
  - Magnetic field model needed for precise navigation.
- All above are often part of the Inertial Measurement Unit (IMU).
- All above need calibration.
- All above benefit from temperature stabilization.

UAV Attitude estimation

Rotation+translation Dynamics model

$\boldsymbol{\tau} = \mathbf{J}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \mathbf{J}\boldsymbol{\omega}$ (29)
$\dot{\mathbf{R}} = \mathbf{R}\boldsymbol{\Omega}$ (30)
$\ddot{\mathbf{r}}^{\mathcal{W}} = \frac{1}{m}\mathbf{R}\mathbf{F}_t^{\mathcal{B}} + \mathbf{g}^{\mathcal{W}}$ (31)

What do we need?
- Angular velocity: $\boldsymbol{\omega}^{\mathcal{B}}$.
- Orientation: $\mathbf{R}$.

State estimator
- a) Complementary filter
- b) EKF
- Often based on quaternions.
- Bias estimation for all the sensors.

UAV Onboard Sensors
- Gyroscope:
  - 3-axis MEMS.
  - **Measured intrinsic angular rate**.
  - Sufficient for attitude rate control.
- Accelerometer:
  - 3-axis MEMS.
  - **Measures proper acceleration**.
  - Gravity model might be needed for precise navigation.
- "Magnetometer":
  - 3-axis
  - **Measures external magnetic field**.
  - Magnetic field model needed for precise navigation.
- All above are often part of the Inertial Measurement Unit (IMU).
- All above need calibration.
- All above benefit from temperature stabilization.

Odometry

Etymology

Odo-metry = Measuring of steps → measuring of where we are based on the steps we took.

Ground robot analogy

- Ground robots have encoders in theirs wheels.
- Encoders' outputs represent intrinsic velocity.
- Integrating encoders from the last known position is called **dead reckoning**.

Can we do *odometry* using the IMU

- Not in general: double-integration of acceleration will drift with increasing velocity.
- Can be done with very precise instruments and models: in aerospace.
- Definitely not with the consumer-level sensors in most UAVs: would not lead to a stable flight.

How can we do odometry then?

- We need to go derivative higher from acceleration: to **velocity**.

Odometry
Etymology
Odo-metry = Measuring of steps → measuring of where we are based on the steps we took.
Ground robot analogy
• Ground robots have encoders in theirs wheels.
• Encoders' outputs represent intrinsic velocity.
• Integrating encoders from the last known position is called dead reckoning.
Can we do odometry using the IMU
• Not in general: double-integration of acceleration will drift with increasing velocity.
• Can be done with very precise instruments and models: in aerospace.
• Definitely not with the consumer-level sensors in most UAVs: would not lead to a stable flight.
How can we do odometry then?
• We need to go derivative higher from acceleration: to velocity.

- To be complete: double integration of acceleration will lead to quadratic drift in position, if the accelerometer exhibits non-zero bias.

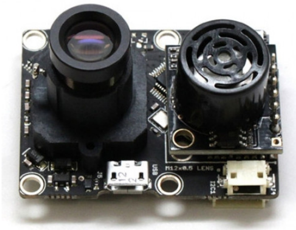
Odometry on UAVs — Optical Flow

Optical flow

- Means of calculating velocity from RGB camera footage.
- Downwards-facing camera.
- Requires distance measurement to fix the absolute velocity.
- Very common on most commercial platforms.
- Parrot AR Drone (2010)
- Relatively robust.
- Not very accurate.

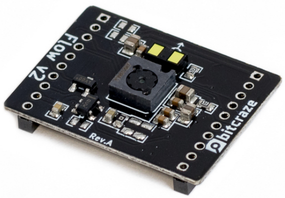
PX4 Flow

- Ultrasound rangefinder
- GreyScale camera
- Embedded μcontroller



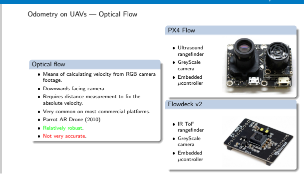
Flowdeck v2

- IR ToF rangefinder
- GreyScale camera
- Embedded μcontroller

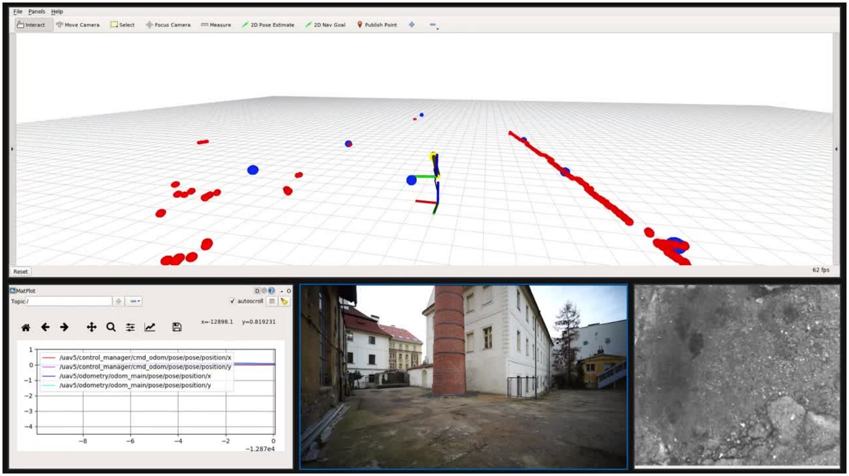


Lecture 3: UAV localization
└─Odometry
└─Odometry on UAVs — Optical Flow

2024-10-07


Odometry on UAVs — Optical Flow
PX4 Flow
Ultrasound rangefinder
GreyScale camera
Embedded μcontroller
Optical flow
Means of calculating velocity from RGB camera footage.
Downwards-facing camera.
Requires distance measurement to fix the absolute velocity.
Very common on most commercial platforms.
Parrot AR Drone (2010)
Relatively robust.
Not very accurate.
Flowdeck v2
IR ToF rangefinder
GreyScale camera
Embedded μcontroller

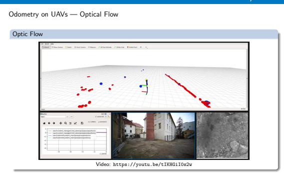
Optic Flow



Video: `https://youtu.be/tIKHGiI0s2w`

2024-10-07

Lecture 3: UAV localization
└─Odometry
└─Odometry on UAVs — Optical Flow



Visual Inertial Odometry

- Combination of feature matching and IMU predictions.
- Does not require a rangefinder.
- Requires proper camera calibration.
- Requires high-resolution and high-rate cameras.
- Global shutter is necessary.
- Robustness is still to be desired (for UAVs).

Features detectors:

- Invariance in transformations and lighting.
- Edges, Corners, Blobs.
- SURF, FAST, SIFT, MSER (Matas et al. [4]).

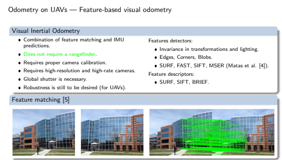
Feature descriptors:

- SURF, SIFT, BRIEF.

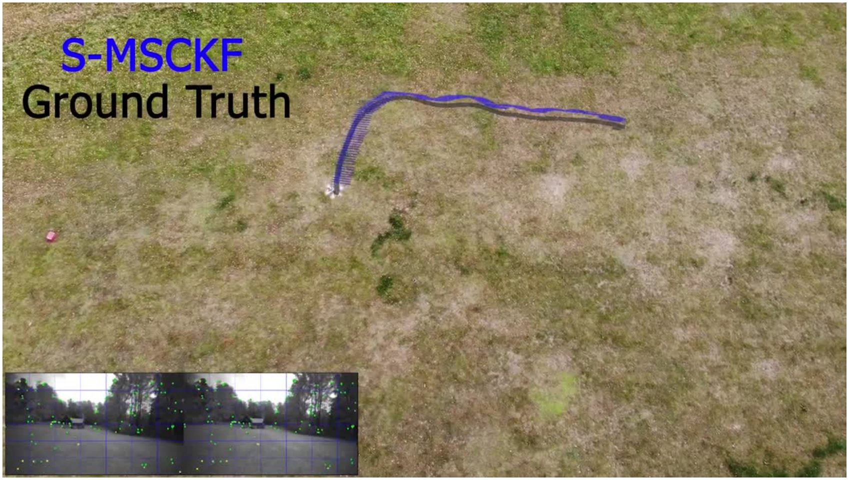
Feature matching [5]

Lecture 3: UAV localization
└─Odometry
└─Odometry on UAVs — Feature-based visual odometry

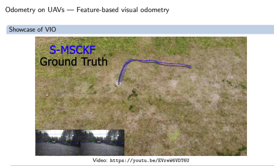
2024-10-07



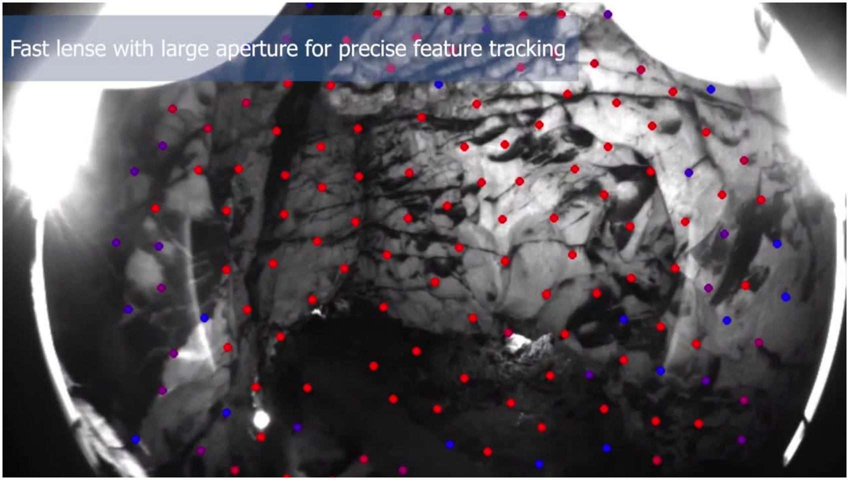
Showcase of VIO



Video: `https://youtu.be/EVreW6VDT6U`

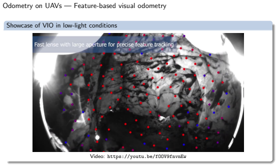


Showcase of VIO in low-light conditions



Video: `https://youtu.be/f0OV9fnvnEw`

Lecture 3: UAV localization
Odometry
Odometry on UAVs — Feature-based visual odometry
2024-10-07



LiDAR

- 2D or 3D.
- Active sensor: Infra-red.
- Scans the environment in stacked rings.
- Has mechanical parts.
- Requires obstacles to be close.

PointCloud data structure

- Organized/unorganized list of 3D points.
- Can contain meta information (reflectivity, color).

PointCloud features

- 3D corners, 3D edges.
- Facets of polyhedra.

Ouster LiDAR Field of View

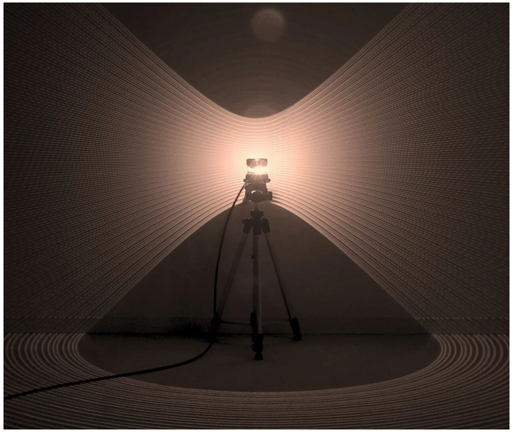


Figure 1: source: `http://ouster.com`

Odometry on UAVs — LiDAR odometry
LiDAR
• 2D or 3D.
• Active sensor: Infra-red.
• Scans the environment in stacked rings.
• Has mechanical parts.
• Requires obstacles to be close.
PointCloud data structure
• Organized/unorganized list of 3D points.
• Can contain meta information (reflectivity, color).
PointCloud features
• 3D corners, 3D edges.
• Facets of polyhedra.
Ouster LiDAR Field of View

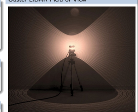
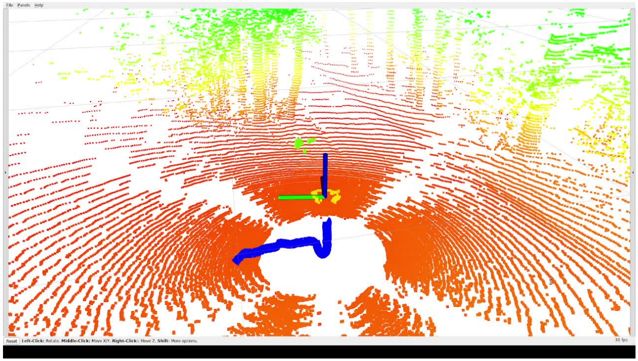


Figure 1: source: http://ouster.com

Iterative Closest Point (ICP)

- Pointcloud registration method.
- Assumption: Not that many points have changed between two consecutively-measured point clouds.
- Minimizing sum of squares of the closest points on two pointclouds.
- Many variants and implementations exist.
- Outlier rejection is important.
- Algorithm:
  1. compute point-to-point correspondences,
  2. optimize for the rotation and translation,
  3. move the pointcloud,
  4. repeat.

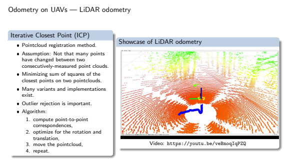
Showcase of LiDAR odometry



Video: `https://youtu.be/veBnoqIqPZQ`

2024-10-07

Lecture 3: UAV localization
└─Odometry
└─Odometry on UAVs — LiDAR odometry



Localization

The means of obtaining the 3D position of the robot in the world coordinate fame.

Why?

- Global localization is needed for global navigation:
  - for building accurate 3D maps of the environment,
  - for using the maps for navigation.
- Localization is needed for any meaningful interaction of a robot with its world.

Where is the state of the art?

- Depends heavily on the use case.

Localization

Localization
The means of obtaining the 3D position of the robot in the world coordinate fame.

Why?
- Global localization is needed for global navigation:
  - for building accurate 3D maps of the environment,
  - for using the maps for navigation.
- Localization is needed for any meaningful interaction of a robot with its world.

Where is the state of the art?
- Depends heavily on the use case.

Global outdoor UAV Localization — GNSS

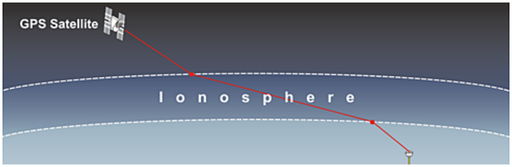
Global Navigation Satellite System

- Earth's satellite constellations.
- GPS, GLONAS, Galileo, BeiDou.

Properties

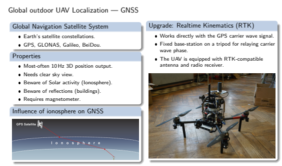
- Most-often 10 Hz 3D position output.
- Needs clear sky view.
- Beware of Solar activity (Ionosphere).
- Beware of reflections (buildings).
- Requires magnetometer.

Influence of ionosphere on GNSS



Upgrade: Realtime Kinematics (RTK)

- Works directly with the GPS carrier wave signal.
- Fixed base-station on a tripod for relaying carrier wave phase.
- The UAV is equipped with RTK-compatible antenna and radio receiver.



- Sensitive to El-Mag interference (USB 3.0).

Global indoor UAV Localization — Motion capture

Marker-based localization

- Pre-set IR camera system.
- IR lighting.
- Retro-reflective markers.
- Popular for control theory research.

Qualisys motion capture cameras

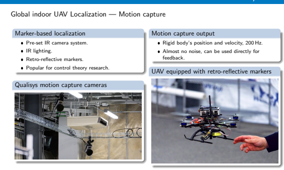
Motion capture output

- Rigid body's position and velocity, 200 Hz.
- Almost no noise, can be used directly for feedback.

UAV equipped with retro-reflective markers

2024-10-07

Lecture 3: UAV localization
└─Localization
  └─Global localization
    └─Global indoor UAV Localization — Motion capture


Global indoor UAV Localization — Motion capture
Marker-based localization
Pre-set IR camera system.
IR lighting.
Retro-reflective markers.
Popular for control theory research.
Motion capture output
Rigid body's position and velocity, 200 Hz.
Almost no noise, can be used directly for feedback.
UAV equipped with retro-reflective markers
Qualisys motion capture cameras

Vijay Kumar's TED talk

Video: `https://youtu.be/4ErEBkj_3PY`

Lecture 3: UAV localization
└─Localization
└─Global localization
└─Global indoor UAV Localization — Motion capture

2024-10-07

Global UAV Localization — Indoor GPS

Ultrasound beacons

- Pre-set environment with ultrasound beacons.
- Known beacon locations.

Radio beacons

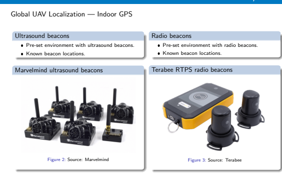
- Pre-set environment with radio beacons.
- Known beacon locations.

Marvelmind ultrasound beacons

Figure 2: Source: Marvelmind

Terabee RTPS radio beacons

Figure 3: Source: Terabee



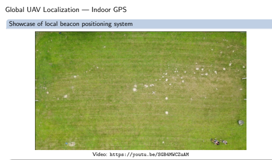
- These systems are very unreliable and are more suitable for ground vehicles.
- Ground vehicles do not need constant precise localization for stabilization, therefore, they cope much better with measurement outages than UAVs.

Showcase of local beacon positioning system

Video: `https://youtu.be/SGB4MWCZuAM`

Lecture 3: UAV localization
└─Localization
  └─Global localization
    └─Global UAV Localization — Indoor GPS

2024-10-07



SLAM — Simultaneous Localization and Mapping

- Creating a map of a priori unknown environment while being localized in the same map.
- *Chicken-and-egg* problem.
- *The holy grail* problem in mobile robotics.
- Two *options*:
  - Online SLAM — computes the current robot pose.
  - Full SLAM — recovers the whole history of the robot poses.

Popular approaches

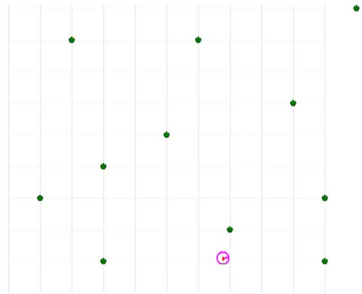
- EKF SLAM,
- Fast SLAM (Particle filter),
- PoseGraph SLAM (Bundle Adjustment),
- Factor Graph SLAM.

Local UAV Localization — SLAM
SLAM — Simultaneous Localization and Mapping
- Creating a map of a priori unknown environment while being localized in the same map.
- Chicken-and-egg problem.
- The holy grail problem in mobile robotics.
- Two options:
  - Online SLAM — computes the current robot pose.
  - Full SLAM — recovers the whole history of the robot poses.
Popular approaches
- EKF SLAM,
- Fast SLAM (Particle filter),
- PoseGraph SLAM (Bundle Adjustment),
- Factor Graph SLAM.

EKF SLAM

- Online SLAM.
- The first SLAM solution, now mostly history.
- The LKF state vector contains:
  - The robot's state $(r_{\mathrm{x}}, r_{\mathrm{y}}, r_{\eta})$,
  - The map of landmarks $(l_{\mathrm{x},n}, l_{\mathrm{y},n})$.
- Assumption: landmark association is solved.
- Capable of loop closure (revisiting places should help).
- Computationally intractable for large maps.

2D EKF SLAM Illustration

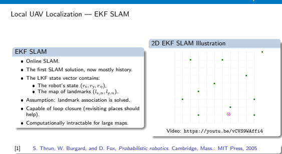


Video: `https://youtu.be/vCVS9WAffi4`

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. Cambridge, Mass.: MIT Press, 2005

2024-10-07

Lecture 3: UAV localization
└─Localization
 └─Local localization
  └─Local UAV Localization — EKF SLAM

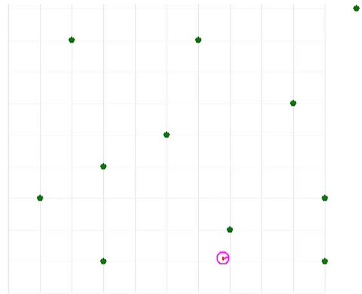


Local UAV Localization — EKF SLAM

Algorithm

1. The state vector and the map are initialized.
2. Prediction step:
   - the robot moves,
   - landmarks are static.
3. Calculation of the expected measurement: which landmarks should be observed and where.
4. Measurement: landmark association.
5. Correction step.
6. Repeat.

2D EKF SLAM Illustration



Video: `https://youtu.be/vCVS9WAffi4`

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. Cambridge, Mass.: MIT Press, 2005

2024-10-07

Lecture 3: UAV localization
└─Localization
└─Local localization
└─Local UAV Localization — EKF SLAM

Local UAV Localization — EKF SLAM
Algorithm
1. The state vector and the map are initialized.
2. Prediction step:
• the robot moves,
• landmarks are static.
3. Calculation of the expected measurement: which landmarks should be observed and where.
4. Measurement: landmark association.
5. Correction step.
6. Repeat.
2D EKF SLAM Illustration
Video: https://youtu.be/vCVS9WAffi4
[1] S. Thrun, W. Burgard, and D. Fox, Probabilistic robotics. Cambridge, Mass.: MIT Press, 2005

Particle filter

- Monte-Carlo localization method.
- Many *particles* representing the robot's model.
- Does not need landmark association.
- The robot's state hypothesis is statistically drawn from the set of particles.

Particle filter — Illustration

- The initial distribution of the particles is random (uniform).
- The robot recognizes a door, but it does not know which door is it.

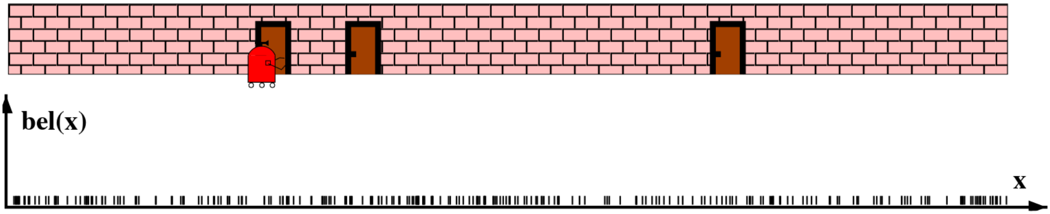
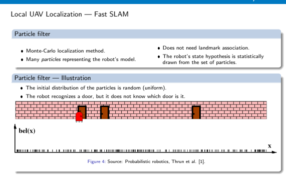


Figure 4: Source: Probabilistic robotics, Thrun et al. [1].



Particle filter — Illustration

- Weight is put to the particles which could generate such measurements.
- Particles move to *next generation*: weighted particles have higher chance to survive and to *multiply*.

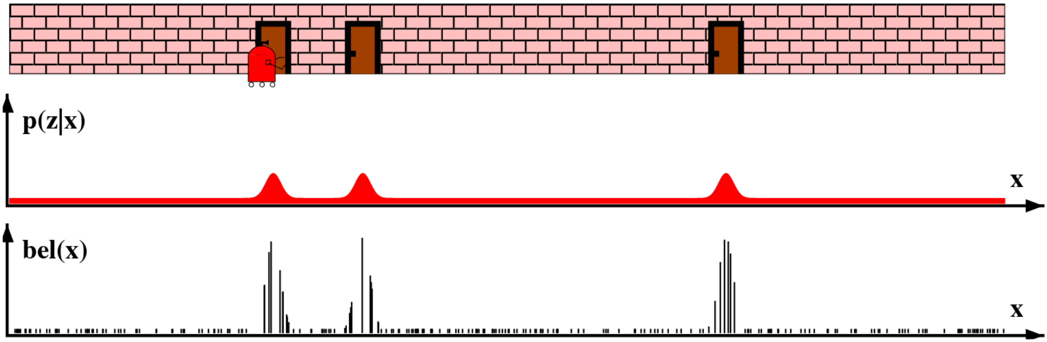
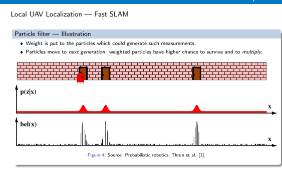


Figure 4: Source: Probabilistic robotics, Thrun et al. [1].



Particle filter — Illustration

- The robot moves in the physical world.
- We apply the control input to each particle and move it as well.

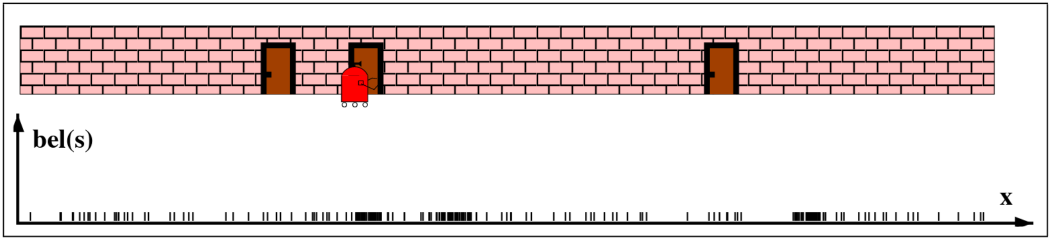
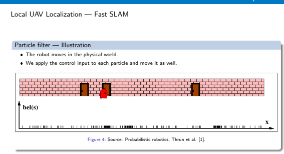


Figure 4: Source: Probabilistic robotics, Thrun et al. [1].

Lecture 3: UAV localization
Localization
Local localization
Local UAV Localization — Fast SLAM

2024-10-07



Particle filter — Illustration

- Robot, again, observes a door, but it does not know which door is it.
- Weight is put to the particles which could generate such measurements.

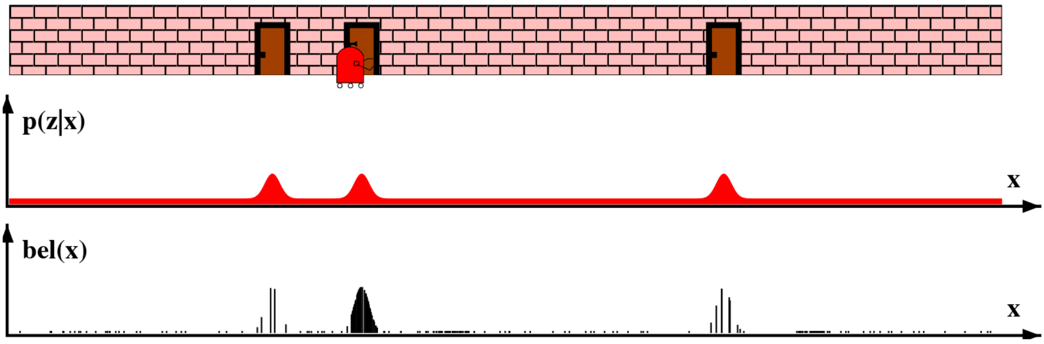
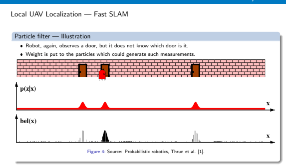


Figure 4: Source: Probabilistic robotics, Thrun et al. [1].



Particle filter — Illustration

- The robot moves in the physical world.
- We apply the control input to each particle and move it as well.

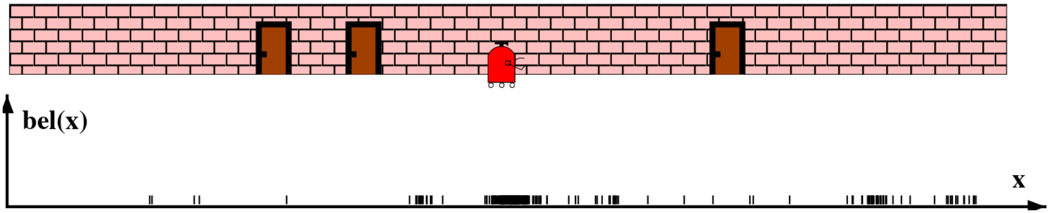
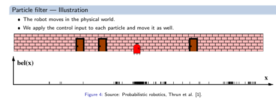
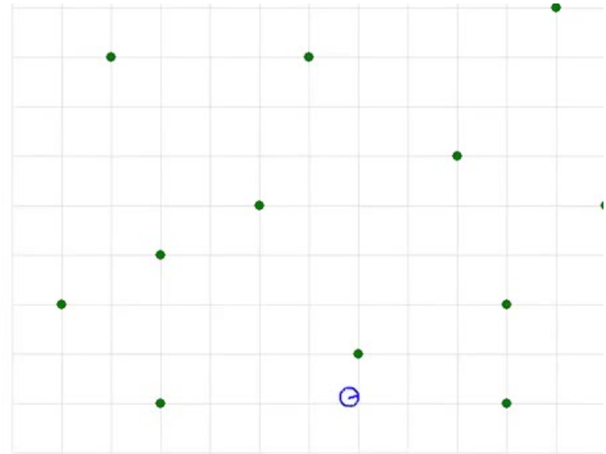


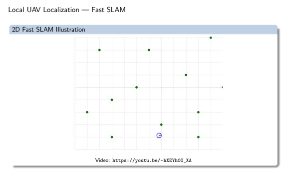
Figure 4: Source: Probabilistic robotics, Thrun et al. [1].



2D Fast SLAM Illustration



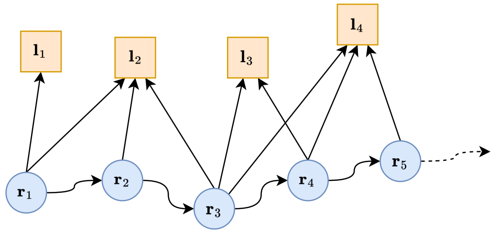
Video: `https://youtu.be/-hXEYhOO_XA`

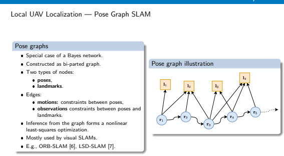


Pose graphs

- Special case of a Bayes network.
- Constructed as bi-parted graph.
- Two types of nodes:
  - **poses**,
  - **landmarks**.
- Edges:
  - **motions:** constraints between poses,
  - **observations** constraints between poses and landmarks.
- Inference from the graph forms a nonlinear least-squares optimization.
- Mostly used by visual SLAMs.
- E.g., ORB-SLAM [6], LSD-SLAM [7].

Pose graph illustration



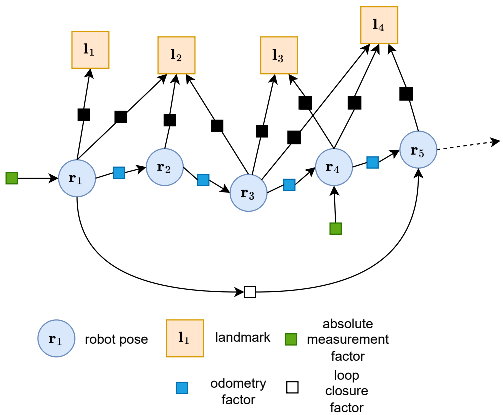


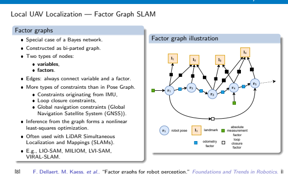
Local UAV Localization — Factor Graph SLAM

Factor graphs

- Special case of a Bayes network.
- Constructed as bi-parted graph.
- Two types of nodes:
  - **variables**,
  - **factors**.
- Edges: always connect variable and a factor.
- More types of constraints than in Pose Graph.
  - Constraints originating from IMU,
  - Loop closure constraints,
  - Global navigation constraints (Global Navigation Satellite System (GNSS)).
- Inference from the graph forms a nonlinear least-squares optimization.
- Often used with LiDAR Simultaneous Localization and Mappings (SLAMs).
- E.g., LIO-SAM, MILIOM, LVI-SAM, VIRAL-SLAM.

Factor graph illustration





Local UAV Localization — Visual SLAMs

List of SOTA Visual SLAM algorithms [9]

Table 2
Summary of topological mapping and localization solutions based on global image descriptors.

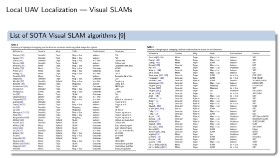
References	Camera	Map	Tasks	Environment	Descriptor
Winters [16]	Omnidir	Topo	Map + Loc	Indoors	PCA
Gaspar [17]	Omnidir	Topo	Map + Loc	Indoors	PCA
Ulrich [18]	Omnidir	Topo	Map + Loc	In + Out	Colour hist.
Werner [46]	Omnidir	Topo	SLAM	Indoors	Colour hist.
Kosecka [19]	Mono	Topo	Map + Loc	Indoors	Gradient orien. hist.
Bradley [20]	Mono	Topo	Map + Loc	Outdoors	WGOH
Weiss [21]	Mono	Topo	Map + Loc	Outdoors	WGII
Wang [22]	Mono	Topo	Map + Loc	In + Out	OACH
Pronobis [23]	Mono	Topo	Loc	Indoors	Receptive field hist.
Singh [48]	Omnidir	Topo	Map + Loc	Outdoors	Gist
Murillo [25]	Omnidir	Hybrid	Map + Loc	In + Out	Omni-gist
Rituerto [49]	Omnidir	Topo	Mapping	Indoors	Omni-gist
Sunderhauf [26]	Mono	Topo	SLAM	Outdoors	BRIEF-gist
Arroyo [53]	Omnidir	Topo	Map + Loc	Outdoors	LDB
Arroyo [55]	Stereo	Topo	Map + Loc	Outdoors	D-LDB
Liu [50]	Mono	Topo	SLAM	Outdoors	Gist
Chapoulie [51]	Sphere	Topo	Map + Loc	In + Out	Gist
Chapoulie [27]	Sphere	Topo	Map + Loc	In + Out	Spherical harmonics
Lamon [28]	Omnidir	Topo	Loc	Indoors	Fingerprints
Tapus [56,57]	Omnidir	Topo	Map + Loc	Indoors	Fingerprints
Liu [29]	Omnidir	Topo	Mapping	Indoors	FACT
Liu [30]	Omnidir	Topo	Mapping	Indoors	DP-FACT
Menegatti [31,32]	Omnidir	Topo	Map + Loc	Indoors	Fourier signatures
Payá [58]	Omnidir	Topo	Map + Loc	Indoors	Fourier signatures
Ranganathan [59]	Omnidir	Topo	Mapping	Indoors	Fourier signatures
Milford [60]	Mono	Hybrid	SLAM	Indoors	Colour segmentation
Prasser [61]	Omnidir	Hybrid	SLAM	Outdoors	Colour hist.
Milford [34]	Mono	Hybrid	SLAM	Outdoors	Scan intensity prof.
Glover [62]	Mono	Hybrid	SLAM	Outdoors	Scan intensity prof.
Lui [36,37]	Omnidir	Hybrid	SLAM	In + Out	2D Haar wavelet dec.
Badino [38]	Mono	Hybrid	Map + Loc	Outdoors	WI-SURF
Xu [64]	Mono	Hybrid	Map + Loc	Outdoors	WI-SURF
Lategahn [39]	Mono	Hybrid	SLAM	Outdoors	DIRD
Nourani [40]	Mono	Topo	Map + Loc	In + Out	OFM/OFSC
Milford [35,65,66]	Mono	Topo	SLAM	Outdoors	Normalized patches
Pepperell [67]	Mono	Topo	SLAM	Outdoors	Normalized patches
Wu [68]	Mono	Topo	Map + Loc	Outdoors	Binarized patches

Table 5
Summary of topological mapping and localization solutions based on local features.

References	Camera	Map	Tasks	Environment	Feature
Kosecka [97–99]	Mono	Topo	Map + Loc	Indoors	SIFT
Zhang [100]	Mono	Topo	Map + Loc	Indoors	SIFT
Zhang [101]	Mono	Topo	SLAM	Indoors	SIFT
Rybski [102]	Omnidir	Topo	Map + Loc	Indoors	KLT
He [103]	Mono	Topo	Map + Loc	Outdoors	SIFT
Sabatta [104]	Omnidir	Topo	Map + Loc	Indoors	SIFT
Johns [105]	Mono	Topo	Map + Loc	Indoors	SIFT
Kawewong [106,107]	Omnidir	Topo	SLAM	In + Out	PIRF (SIFT)
Tongprasit [108]	Omnidir	Topo	SLAM	In + Out	PIRF (SURF)
Morioka [109]	Omnidir	Hybrid	SLAM	Indoors	3D-PIRF (SURF)
Andreasson [90]	Omnidir	Topo	Map + Loc	Indoors	KLT/M-SIFT
Valgren [110]	Omnidir	Topo	Mapping	Indoors	KLT/M-SIFT
Valgren [111]	Omnidir	Topo	Mapping	In + Out	SIFT
Valgren [112]	Omnidir	Topo	Loc	Outdoors	SIFT/SURF
Ascani [113]	Omnidir	Topo	Loc	In + Out	SIFT/SURF
Anati [114]	Omnidir	Topo	Map + Loc	In + Out	SIFT
Zivkovic [115]	Omnidir	Hybrid	Map + Loc	Indoors	SIFT
Booij [116]	Omnidir	Hybrid	Map + Loc	Indoors	SIFT
Booij [117]	Omnidir	Hybrid	Map + Loc	In + Out	SIFT
Dayoub [118]	Omnidir	Hybrid	Map + Loc	Indoors	SURF
Blanco [119,120]	Stereo	Hybrid	SLAM	Indoors	SIFT
Tully [121]	Omnidir	Hybrid	Map + Loc	Indoors	SIFT
Tully [122]	Omnidir	Hybrid	SLAM	Indoors	SIFT
Segvic [123]	Mono	Hybrid	Map + Loc	Outdoors	SIFT/Harris/MSER
Ramisa [124]	Omnidir	Topo	Map + Loc	Indoors	MSER/SIFT/GLOH
Badino [125]	Mono	Hybrid	Map + Loc	Outdoors	SURF/U-SURF
Dayoub [126]	Omnidir	Topo	Map + Loc	Indoors	SURF
Bacca [127,128]	Omnidir	Topo	Map + Loc	Indoors	SIFT/SURF
Bacca [129]	Omnidir	Topo	SLAM	Indoors	Edges
Romero [130,131]	Omnidir	Topo	SLAM	Outdoors	MSER
Majdik [132]	Mono	Topo	Loc	Outdoors	ASIFT
Saedan [133]	Omnidir	Hybrid	SLAM	Indoors	Wavelets
Kessler [134]	Omnidir	Topo	SLAM	Indoors	SIFT
Maohai [135]	Omnidir	Topo	Map + Loc	Indoors	ASIFT
Garcia-Fidalgo [136]	Mono	Topo	SLAM	In + Out	SURF
Garcia-Fidalgo [137]	Mono	Topo	SLAM	In + Out	SIFT

2024-10-07

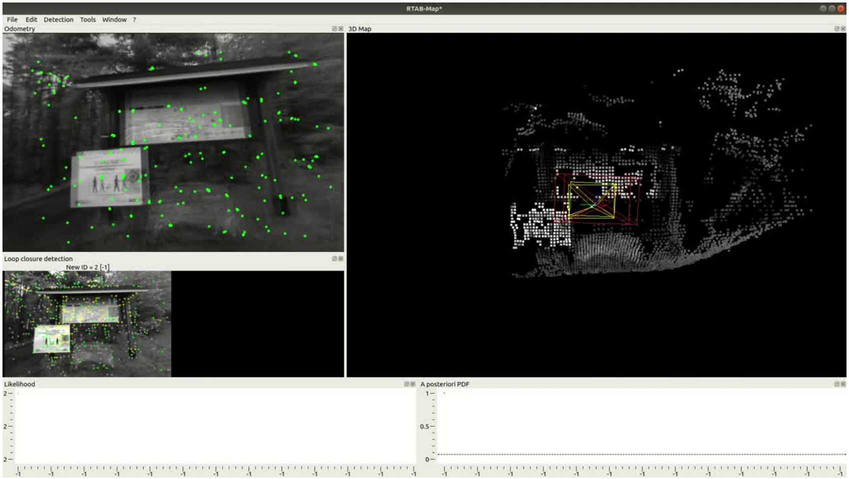
Lecture 3: UAV localization
└ Localization
└ Local localization
└ Local UAV Localization — Visual SLAMs



- The research field of visual SLAMs is huge and also very popular.
- Almost everyone can contribute, because you only need a camera to start working.
- Rarely anything works in the real world and onboard a UAV.

Local UAV Localization — Visual SLAMs

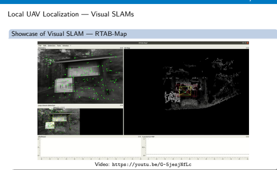
Showcase of Visual SLAM — RTAB-Map



Video: `https://youtu.be/G-5jesjNfLc`

2024-10-07

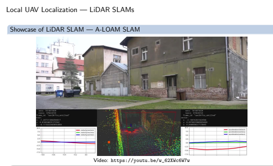
Lecture 3: UAV localization
└─Localization
  └─Local localization
    └─Local UAV Localization — Visual SLAMs



- The video showcases RGBD SLAM.
- RTAB-Map can also utilize other sources of data to build a map, e.g., LiDAR pointclouds.

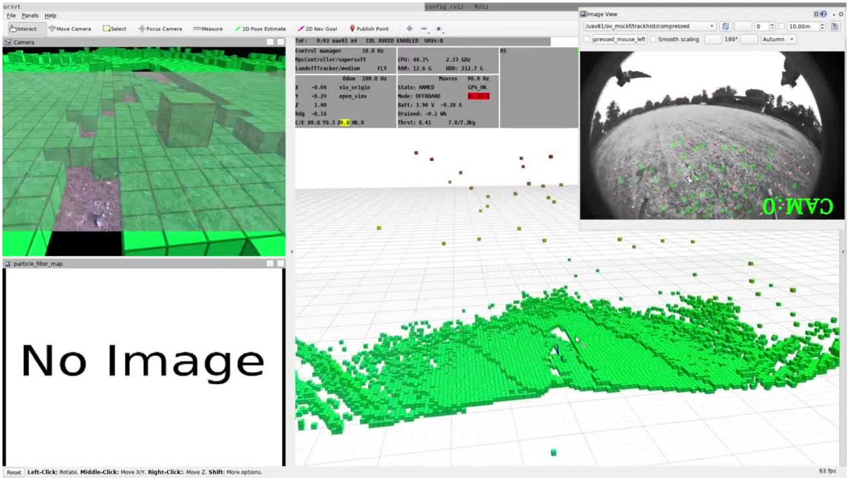
Showcase of LiDAR SLAM — A-LOAM SLAM

Video: `https://youtu.be/w_62XWc6W7w`

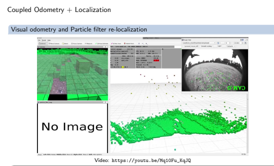


- The field of LiDAR SLAMs is also very active and rich.
- Similarly, true SLAMs are rarely used on UAVs, mostly due to the SLAMs' computational demands.

Visual odometry and Particle filter re-localization



Video: `https://youtu.be/Mq10Fu_KqJQ`



- Open-VINS odometry for fast state estimation.
- Particle filter for re-localization in a known height map.

Localization — Summary

- UAVs most commonly operate outdoors, therefore, **GNSS localization the most common**.
- Commercial platforms are capable of onboard odometry (most often visual), however, that is used for **stabilization** and to aid human pilots with control in GNSS-denied environments.
- SLAMs are mostly the **subject of research** and are not reliable enough to use the UAVs to their full potential.
- Multi-modal SLAMs and geometries are probably the future. Fusion of different sensor modalities (Visual, LiDAR, Radar, InfraRed) will increase the reliability and robustness.

Localization — Summary
- UAVs most commonly operate outdoors, therefore, **GNSS localization the most common**.
- Commercial platforms are capable of onboard odometry (most often visual), however, that is used for **stabilization** and to aid human pilots with control in GNSS-denied environments.
- SLAMs are mostly the **subject of research** and are not reliable enough to use the UAVs to their full potential.
- Multi-modal SLAMs and geometries are probably the future. Fusion of different sensor modalities (Visual, LiDAR, Radar, InfraRed) will increase the reliability and robustness.

References

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. Cambridge, Mass.: MIT Press, 2005.

[2] E. A. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," in *Adaptive Systems for Signal Processing, Communications, and Control Symposium, IEEE*, IEEE, 2000, pp. 153–158.

[3] T. Baca, P. Stepan, B. Spurny, D. Hert, R. Penicka, M. Saska, J. Thomas, G. Loianno, and V. Kumar, "Autonomous Landing on a Moving Vehicle with an Unmanned Aerial Vehicle," *Journal of Field Robotics*, vol. 36, pp. 874–891, 5 2019.

[4] S. Obdrzalek and J. Matas, "Object recognition using local affine frames on maximally stable extremal regions," in *Toward category-level object recognition*, Springer, 2006, pp. 83–104.

[5] S. A. K. Tareen and Z. Saleem, "A comparative analysis of sift, surf, kaze, akaze, orb, and brisk," in *2018 International conference on computing, mathematics and engineering technologies (iCoMET)*, IEEE, 2018, pp. 1–10.

[6] C. Campos, R. Elvira, J. J. G. Rodriguez, J. M. Montiel, and J. D. Tardos, "ORB-SLAM3: An accurate open-source library for visual, visual-inertial, and multimap SLAM," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[7] J. Engel, T. Schops, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *European conference on computer vision*, Springer, 2014, pp. 834–849.

[8] F. Dellaert, M. Kaess, *et al.*, "Factor graphs for robot perception," *Foundations and Trends in Robotics*, vol. 6, no. 1-2, pp. 1–139, 2017.

[9] E. Garcia-Fidalgo and A. Ortiz, "Vision-based topological mapping and localization methods: A survey," *Robotics and Autonomous Systems*, vol. 64, pp. 1–20, 2015.

2024-10-07

Lecture 3: UAV localization
└─Localization
└─Local localization
└─References

References

[1] S. Thrun, W. Burgard, and D. Fox, Probabilistic robotics. Cambridge, Mass.: MIT Press, 2005.
[2] E. A. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," in Adaptive Systems for Signal Processing, Communications, and Control Symposium, IEEE, IEEE, 2000, pp. 153–158.
[3] T. Baca, P. Stepan, B. Spurny, D. Hert, R. Penicka, M. Saska, J. Thomas, G. Loianno, and V. Kumar, "Autonomous Landing on a Moving Vehicle with an Unmanned Aerial Vehicle," Journal of Field Robotics, vol. 36, pp. 874–891, 5 2019.
[4] S. Obdrzalek and J. Matas, "Object recognition using local affine frames on maximally stable extremal regions," in Toward category-level object recognition, Springer, 2006, pp. 83–104.
[5] S. A. K. Tareen and Z. Saleem, "A comparative analysis of sift, surf, kaze, akaze, orb, and brisk," in 2018 International conference on computing, mathematics and engineering technologies (iCoMET), IEEE, 2018, pp. 1–10.
[6] C. Campos, R. Elvira, J. J. G. Rodriguez, J. M. Montiel, and J. D. Tardos, "ORB-SLAM3: An accurate open-source library for visual, visual-inertial, and multimap SLAM," IEEE Transactions on Robotics, vol. 37, no. 6, pp. 1874–1890, 2021.
[7] J. Engel, T. Schops, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in European conference on computer vision, Springer, 2014, pp. 834–849.
[8] F. Dellaert, M. Kaess, et al., "Factor graphs for robot perception," Foundations and Trends in Robotics, vol. 6, no. 1-2, pp. 1–139, 2017.
[9] E. Garcia-Fidalgo and A. Ortiz, "Vision-based topological mapping and localization methods: A survey," Robotics and Autonomous Systems, vol. 64, pp. 1–20, 2015.

Thanks for listening.

Conclusion

Thanks for listening.